HACK A ZIP-ZAP • ROBOGAMES • ROBOTS WHO SPEAK

SERVO MAGAZINE

FOR THE ROBOT EXPERIME

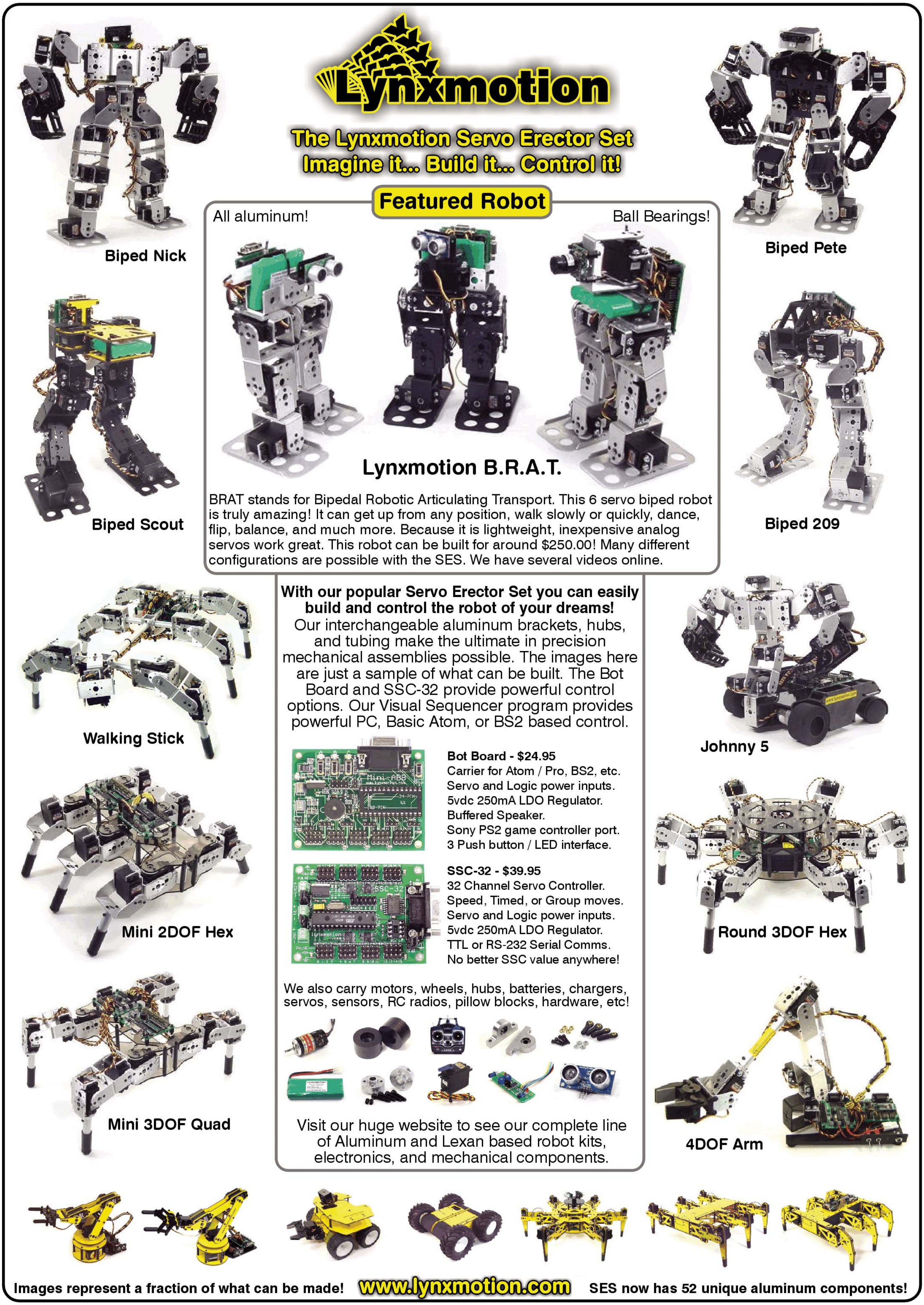
www.servomagazine.com
AUGUST 2006

Does Your Robot Lack Personality?

Give Him/Her a Facelift!

- Soldering for Robotics
- The Importance of Being Educated
- How to Let Your Robot See

U.S. $5.50 CANADA $7.00
08>
0 74470 58285 4


Lynxmotion
The Lynxmotion Servo Erector Set
Imagine it... Build it... Control it!
Featured Robot
All aluminum!
Ball Bearings!
Lynxmotion B.R.A.T.
BRAT stands for Bipedal Robotic Articulating Transport. This 6 servo biped robot is truly amazing! It can get up from any position, walk slowly or quickly, dance, flip, balance, and much more. Because it is lightweight, inexpensive analog servos work great. This robot can be built for around $250.00! Many different configurations are possible with the SES. We have several videos online.
Biped Nick
Biped Pete
Biped Scout
Biped 209
With our popular Servo Erector Set you can easily build and control the robot of your dreams!
Our interchangeable aluminum brackets, hubs, and tubing make the ultimate in precision mechanical assemblies possible. The images here are just a sample of what can be built. The Bot Board and SSC-32 provide powerful control options. Our Visual Sequencer program provides powerful PC, Basic Atom, or BS2 based control.
Walking Stick
Johnny 5
Bot Board - $24.95
Carrier for Atom / Pro, BS2, etc.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
Buffered Speaker.
Sony PS2 game controller port.
3 Push button / LED interface.
SSC-32 - $39.95
32 Channel Servo Controller.
Speed, Timed, or Group moves.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
TTL or RS-232 Serial Comms.
No better SSC value anywhere!
Mini 2DOF Hex
Round 3DOF Hex
We also carry motors, wheels, hubs, batteries, chargers, servos, sensors, RC radios, pillow blocks, hardware, etc!
Mini 3DOF Quad
4DOF Arm
Visit our huge website to see our complete line of Aluminum and Lexan based robot kits, electronics, and mechanical components.
Images represent a fraction of what can be made!
www.lynxmotion.com
SES now has 52 unique aluminum components!

SERVO

Coming 09.2006

WIRELESS COMMUNICATIONS FOR TELE-AUTONOMOUS ROBOTS

Features & Projects

SERVO Magazine **(ISSN 1546-0592/CDN Pub Agree#40702530)** is published monthly for $24.95 per year by T & L Publications, Inc., 430 Princeland Court, Corona, CA 92879. PERIODICALS POSTAGE PAID AT CORONA, CA AND AT ADDITIONAL ENTRY MAILING OFFICES. POSTMASTER: Send address changes to **SERVO Magazine, P.O. Box 15277, North Hollywood, CA 91615** or Station A, P.O. Box 54, Windsor ON N9A 6J5; cpcreturns@servomagazine.com

08.2006
VOL. 4 NO. 8

Departments

ENTER WITH CAUTION!

Columns

ON THE COVER

Mind / Iron

by Bryce and Evan Woolley

A FIRST Rate Education

Although FIRST Robotics is a program that has shaped our high school experiences, and the experiences of thousands of young aspiring minds across the world, the journey does not stop at graduation from high school. In one sense, the journey of FIRST is that of learning and inspiration, and an inquisitive mind never stops learning. But even in the very literal sense, FIRST has had a real impact on our college experiences at the beautiful and illustrious University of California, San Diego. And our experiences have been very different at that, since Evan is pursuing a major in the university's rigorous Mechanical and Aerospace Engineering department, while Bryce is immersed in the social sciences with his chosen path of study encompassing sociology, social psychology, political science, and history. But regardless of the path one chooses to take, in college and beyond, the knowledge acquired through participation in FIRST, the pillars of gracious professionalism, and the doors this program opens all lead to a brighter and more inspired future.

For me (Evan), FIRST robotics has had very tangible consequences in furthering my studies in engineering at the university level. In my mechanical engineering and physics courses, I often think back to my experiences in FIRST as practical examples of the theoretical groundwork laid in my classes. When my Physics 2A professor lectured about torque, I reminisced about doing some calculations for Dreimo's robotic arm during the 2005 FIRST season, and when my MAE 1 professor lectured about the center of gravity, I recalled the design considerations from the recent 2006 FIRST season.

FIRST offers an infinitely valuable context of understanding for engineering students. The concepts learned in every engineering related class I have taken so far deal with things that I have already been introduced to in FIRST. Everything from C programming to quasi-static analysis is encompassed in the omnibus of engineering that is the FIRST Robotics Competition, and I can say without a doubt that my early introduction to these fields — fields that are otherwise so esoteric in our culture today — has greatly accelerated my understanding at the university level. I really feel like it has made me a better engineering student, and that will undoubtedly make me a better engineer.

My experience in FIRST has also helped me outside of the classroom. To help with the difficult task of paying for a university education, I applied for and received a generous scholarship (available only to graduating FIRST students) from Raytheon. I also received a scholarship from International Rectifier, which although is not only available to FIRST alumni, was undoubtedly awarded in part in recognition of the growth and involvement I experienced with FIRST.

As a scholarship recipient from IR, I was also offered a position as a lab intern for this summer, which I graciously accepted and am currently working as. These experiences hint at another great opportunity that FIRST offers — the opportunity to make connections. While most people might think this would take years of diligent networking, it's not unusual for FIRST alumni to be able to rightly claim that they have met the Director of Solar System Exploration for NASA (Dave Lavery), the Pappalardo Professor of Mechanical Engineering at MIT (Woodie Flowers), and one of the greatest technological innovators of our time (Dean Kamen).

The chance to talk to the FIRST judges is also a rare opportunity that gives FIRST students face time with the leaders of industry. I already feel that it would be a lot less intimidating applying for jobs after college at places like Northrop-Grumman and Raytheon after getting the chance to talk to some of their actual employees. Overall, FIRST gives prospective engineering students a great head start in technical understanding and valuable networking.

For me (Bryce), FIRST has provided me with a solid foundation of understanding and respect for technology and science in

Mind/Iron Continued →

Published Monthly By
T & L Publications, Inc.
430 Princeland Court
Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Product Order Line **1-800-783-4624**
www.servomagazine.com

Subscriptions
Inside US **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277
North Hollywood, CA 91615

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

ASSOCIATE PUBLISHER/ VP OF SALES/MARKETING
Robin Lemieux
display@servomagazine.com

CONTRIBUTING EDITORS
Dave Prochnow, Tom Carroll, Pete Miles, David Geer, Jack Buffington, R. Steven Rainwater, Gordon McComb, Michael Simpson, Chris Cooper, Kevin Berry, Jeff Eckert, James Isom, Bryan Bergeron, Michael Vroegop, Evan Woolley, Bryce Woolley, Guy Marsden, Russ Barrow, Dave Calkins, Peter Best

CIRCULATION DIRECTOR
Tracy Kerley
subscribe@servomagazine.com

WEB CONTENT/STORE
Michael Kaudze
sales@servomagazine.com

PRODUCTION/GRAPHICS
Shannon Lemieux

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

society, which is important for every contributing individual to have. Although as a social scientist I may not necessarily use the physics of robotic arm extension in my everyday career, principles like gracious professionalism and group collaboration will serve me well in any capacity where I need to work with other people (which as a social scientist is basically unavoidable).

FIRST has also had a very real impact on the advancement of my college career. Not to mention its favorable appearance on my college application, it has helped to accelerate my concentration on my main area of academic interest. While in high school, I took a rigorous load of courses in the hard sciences (like AP physics and chemistry) to give me an intellectual edge in the FIRST competition. FIRST also inspired me to take these tough courses out of interest — I wanted to learn how stuff in the world worked. This inspiration became money in my pocket when my high scores on the $80 AP tests exempted me from comparable $600 courses at the university level. They also fulfilled most of my hard science general education requirements, allowing me to focus on the real area of my academic interest (the social sciences) earlier than other freshmen in the same major, thus accelerating my major coursework.

Additionally, participation in FIRST gave me an insider look at an aspect of sociology that many students in the same major do not have. The study of the social impact of advancing technology and the interplay between science and society is an area of increasing interest in the community of social scientists. When engineers work in labs, they are not just manipulating materials and harnessing physical energy — they are changing the way society thinks and behaves in relation to its environment. And engineers today need not only be aware of the kinematics of the machines they develop, but of the attendant social kinetics of accountability and consequence. The advancing technology developed by engineers today — of whom FIRST alumni are beginning to join the ranks of — is changing society. They are changing the world. **SV**

BIO-FEEDBACK

Dear SERVO:

I'm really enjoying Chris Cooper's Mobility to the Maxx series of articles. I'm about to tackle my first robotics project: a navigating autonomous robot built on an electric-converted Traxxas Revo. Thus, these articles will be invaluable. In future editions, I'd like to see additional sensors and enhanced autonomy.

Michael Gross

Dear SERVO:

I noticed a typo in the source code for my column that is non-obvious with what I turned in last month.

In the PIC C code file, I have the line: This program runs on a PIC16F83 processor with a 20MHz clock. It can be compiled. It should have been "runs on a PIC16F873 processor ... "

Jack Buffington
Rubberbands and Baling Wire

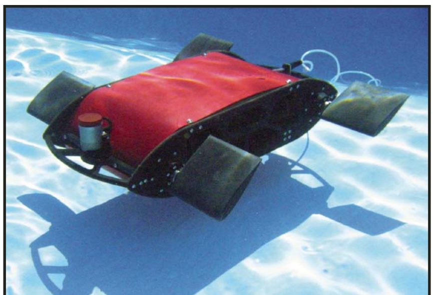
Robytes

by Jeff Eckert

Are you an avid Internet surfer who came across something cool that we all need to see? Are you on an interesting R&D group and want to share what you're developing? Then send me an email! To submit related press releases and news items, please visit www.jkeckert.com

— Jeff Eckert

Bot Helps Understand Flipper Motion



Madeleine — the flippered underwater robot. Photo by John Long, Vassar College.

If you have lain awake at night wondering why four-flippered animals such as penguins, seals, and sea turtles tend to use only two limbs for propulsion, a good night's rest is on the way. Through the efforts of John Long at Vassar College (**www.vassar.edu**), some folks from Nekton Research, LLC (**www.nektonresearch.com**), and some funding from the National Science Foundation (**www.nsf.gov**), a four-flippered robot named Madeleine was created, and she appears to have come up with the answer.

When the joystick-controlled bot was recently plunked into the water and put to the test, it was discovered that her cruise speed did not increase when she used all four flippers instead of just the rear ones, apparently because turbulence created by the front flippers interfered with the ability of the rear ones to generate forward motion. It was also noted that maintaining the same speed with all four required the expenditure of more energy. However, the use of all four limbs for stopping did work better.

It has been speculated that this research will help scientists and engineers figure out more energy-efficient ways to use flippers for locomotion, thus aiding in the design of underwater autonomous vehicles. But another interesting aspect is that, according to scientists who deal with such things, four-limbed aquatic dinosaurs such as plesiosaurs appear to have been built to use all four flippers for movement, leading to speculation that they used all fours to attack prey.

Speed-Reading Robot Creates Archives

The model DL-3000 robotic page turner and scanner. Photo courtesy of 4DigitalBooks – ASSY SA.

An impressive product for libraries and other entities that need to archive huge quantities of documents is the robotic page turner and scanner from a Swiss company, 4DigitalBooks (**www.4digitalbooks.com**). The top-end model — the DL-3000 — can process up to 3,000 pages per hour using twin digital cameras to shoot left- and right-hand pages simultaneously, and it is said to be capable of working 24/7 in unattended operation. Particular care has been taken to create a mechanism that is suitable for working with old, fragile books without damaging them, and it uses a "nonagressive" light source for additional safety.

The machine can also handle magazines and bound newspapers in sizes from A5 to A2 (148 x 210 to 420 x 594 mm, or about 5.8 x 8.25 to 16.5 x 23.4 inches), and it accepts books with mixed paper thickness, texture, and porosity. You aren't likely to tuck it away in a corner of your living room though, as it stands 3.1 x 1.5 x 2.2 m (approx. 10 x 5 x 7 feet) and weighs in at 1,200 kg (3,200 lbs).

The company's goal is a near future in which "a majority of books will be reachable online and where full text search would be possible inside their content." It appears that the 4DigitalBooks visionaries have never heard of copyright laws.

Give Your Vac a Personality

myRoomBud™ offers multiple personalities for the Roomba® floor vacuum.

It is arguable that, when it comes to generating a lot of attention for a marginally interesting product, the Roomba floor vac is unsurpassed. However, it seems impossible to resist mentioning that if you are one of the two million owners, you may be interested in giving it some personality using dress-up and programming kits that are available from myRoomBud (**www.myroombud.com**).

RoomBud Personalities enhance the Roomba pet experience by "teaching" your Roomba to act like the pet or character you choose. For example, Roobit the Frog hops around, Roor the Tiger

growls then pounces, and RoomBette La French Maid wiggles its behind at you before vacuuming your room.

This is accomplished by programming the vac via IRobot Roomba Open Interface and RooTooth, a Bluetooth adaptation, from your PC. Videos of the various personalities — which also include Slops the Pig, FooFoo the WereRabbit, and others — can be viewed at **www.myroombud.com/itsalive.html**

If all of this seems a little juvenile, it's probably because the company is a "profitable, privately owned company started by kids, built by kids, and run by kids." The various personalities will run you $24.95 each, plus $4 shipping.

Robotic Bartender Available

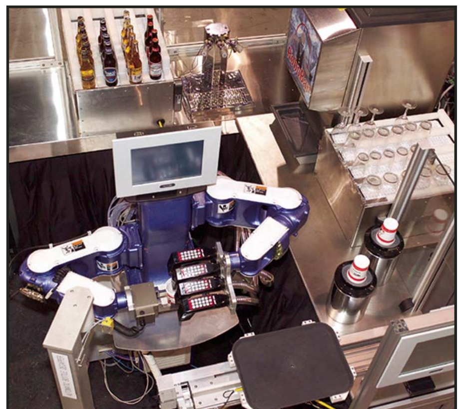
Let's be up front here. There is quite a bit of redundancy in the robotics field, and sometimes it is difficult to find anything to get excited about. One stepper motor looks pretty much like another one, and there is already a mechanical version of nearly everything that walks, crawls, hops, or flies. But once in a while you run into something that pushes the limits, and such is Motoman's new RoboBar™ series of robotic bartending and beverage dispensing systems.

At the top of the line is the HP (high-production) model, which can produce a mixed drink every 10 to 15 seconds. Aimed for use in casinos and other high-volume service bars, it features a dual-arm Motoman DA9IC robot with an NXC100 controller in its base. The arms each have five axes of motion, and the base rotates to provide an 11th axis.

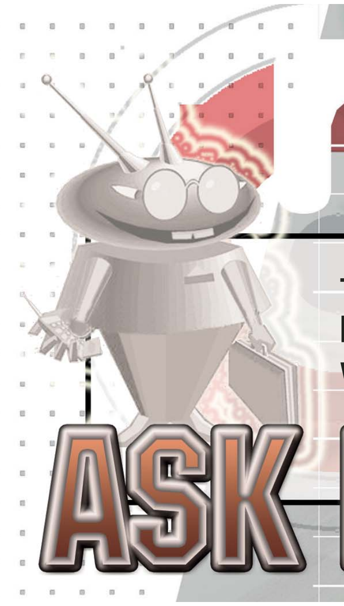
One arm is equipped with a simple parallel jaw gripper that handles cups, glasses, and beer bottles. Up to eight dispensing guns are mounted on the robot's other arm. Each gun can dispense up to 16 different ingredients (128 total), including liquors, mixes, juices, and wines, in any combination. It places multiple drinks onto a tray that is then shuttled in and out of the mixing cell (which includes a safety enclosure).

For lower-volume applications, you can choose the E (entertainment) model, which is equipped with a magnetic card scanner that allows a customer to swipe his card and order a beverage via a touch screen. A flat-panel video screen even allows you to choose a male or female personality for your bartender interface, with a matching voice. The bartender can be programmed to provide information or tell jokes. Finally, Motoman offers the NA (no alcohol) version, which is designed to dispense hot coffee drinks, soft drinks, and other nonalcoholic beverages.

As pointed out by the company, RoboBar can work 24 hours a day, seven days a week, without breaks, vacations, holidays, sick time, or hangovers. It always works at 100 percent efficiency and never asks for a raise. It runs on about 30 cents' worth of electricity per hour. Details on purchasing, leasing, or renting a unit are available at **www.motoman.com** Cheers! **SV**



Motoman is "pioneering drinkmation" with three versions of its robotic bartender. Photo courtesy of Motoman, Inc.



Our resident expert on all things robotic is merely an Email away.

roboto@servomagazine.com

Tap into the sum of *all human knowledge* and get your questions answered here! From software algorithms to material selection, Mr. Roboto strives to meet you where you are — and what more would you expect from a complex service droid?

ASK MR. ROBOTO

by
Pete Miles

Q. I have recently purchased a set of Easy Roller motors from Solutions Cubed because their output shaft makes it easy to add a shaft encoder. They are really nice motors, but I would like to get more torque out of them. I like the size of these motors, and I haven't been able to find any slower, higher torque motors with an external output shaft for an encoder. Do you have any suggestions as to where I can find some inexpensive motors like this?

— Mike Baily

A. Solutions Cubed (**www.solutions-cubed.com**) offers a nice motor option that enables attaching a quadrature encoder directly on the motor's shaft, which enables great speed and position control and accuracy. I have a set of them for the balancing robot that I am currently working on. These motors are manufactured by Hsiang-Neng (**www.hsiangnengmotors.com.tw**), and the general specifications for this motor are listed in Table 1. Figure 1 shows a photo of this motor.

There are basically three different things you can do here. First, change your motors with higher current rated motors. Second, increase the motor's supply voltage and use a closed loop motor speed controller. And third, change the motor's gearbox. Changing the motors with a higher current/power rating (higher current rating usually means a higher torque rating) is what most people do. But in your case, finding the right motors that have an encoder output shaft may prove to be difficult. Since you are using encoders on your motors, I am assuming that you are already implementing some form of a closed loop speed control system in your robot, but since you are asking this question, increasing the motor's supply voltage probably doesn't provide for your needs.

With this particular motor, probably the easiest thing to do is to change your motor's gearbox with one that has a higher gear reduction. Jameco Electronics (**www.jameco.com**) probably has the largest selection of these gear motors available, and Table 2 lists a small selection of their compatible gear motors sorted by increasing gear reduction order. The only motors that

Motor Model No.	HN-GH12-1634TR
Operating Voltage Range	4.5–12V
Gear Ratio	30:1
Internal Resistance	10 ohm
No Load Speed @ 12V	200 RPM
No Load Current @ 12V	111 mA
Max. Efficiency Speed @ 12V	145 RPM
Max. Efficiency Current @ 12V	293 mA
Torque @ Max. Efficiency	850 g-cm
Stall Current	1.2 A
Stall Torque	3.6 kg-cm
Shaft Diameter	6 mm
Shaft Length	18 mm
Motor Diameter	37 mm
Overall Motor Length with Encoder Shaft	54.4 mm
Weight	120 g

Table 1. General specifications for Solutions Cubed Easy Roller motor.

Figure 1. Easy Roller motor from Solutions Cubed.

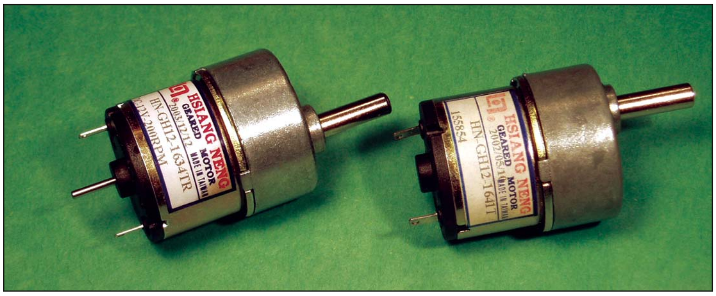


Figure 2. Both 30:1 and 60:1 gear motors prior to switching gearboxes.

Gear Ratio	Jameco Part No.	Hisang-Nang Part No.	Speed (RPM)	Torque (g-cm)	Current (mA)
10:1	253446CK	38-001	600	350	61.8
30:1	161381CK	GH12-1634T	145	850	293
33:1	253462CK	38-003	180	800	68
50:1	253471CK	38-004	120	1100	74
56:1	253489CK	38-005	110	1300	76
60:1	155863CK	GH12-1926Y-F	70	1000	300
90:1	253489CK	38-006	66	2100	80
100:1	253500CK	38-007	60	3200	82
150:1	253518CK	38-008	40	4800	87.5
167:1	253526CK	38-009	36	5400	93.5
250:1	233534CK	38-010	24	7500	97
270:1	155838CK	GH12-1641T-L	15	3000	250
1000:1	155820CK	GH12-1830Y-P	4.5	4500	220
3000:1	155011CK	GH12-1926	2	6000	250

Table 2. Gear motors with interchangeable gearboxes. (Performance data are maximum efficiency specs at 12V.)

have interchangeable gearboxes with the Easy Roller gear motor are the GH35GM and 38GM series gear motors that Jameco sells. The DC motor portion of the GH35GM series motors all have a different winding, so they have different performance specifications under the same operating conditions. The 38GM series motors all use the same DC motor configuration.

The gearbox on the Easy Roller gear motor has a 30:1 gearbox. So if you need to double the torque capabilities of your motor, you will need to double the gear reduction, in this case, a 60:1 gear reduction. Keep in mind, when you double the gear reduction, the output shaft speed will be reduced by the same proportion. Figure 2 shows a photo of this motor and another motor with a 60:1 gearbox that will be used to demonstrate switching gearboxes.

Changing the gearbox is a relatively simple task. There are three screws that are used to hold the gearbox cover to the motor. Remove them and slide off the cover. The shaft may stay in the cover (see Figure 3).

Next, remove the gears from the gearbox — make sure that you remember the order in which you removed them and the order in which you removed the small bronze bushings (see Figure 4).

Next, remove the two screws that hold the base of the gearbox to the face of the motor, and remove the gearbox mounting plate (Figure 5).

You should notice that there are six threaded holes on the face of the motor housing (Figure 6). Only two of them are needed to attach the base of the gearbox to the motor body. As a side note, since the output shaft on the gearbox is offset from the motor's centerline by 0.276 inches (7 mm), the angular orientation of this shaft with respect to orientation of the motor's electrical terminal tabs (at the rear of the motor) can be adjusted in 60-degree increments. This is done by rotating the base of the gearbox around the axle until the output shaft's orientation is at its desired location. In some cases, this orientation is important due to geometrical mounting constraints, but it will have no effect on the overall performance of the motor.

To attach the new gearbox to the Easy Roller motor, repeat these steps in reverse order. The amount of time to do all this only takes about five minutes. When you are done, you will

Figure 3. Removing the cover of the gearboxes.

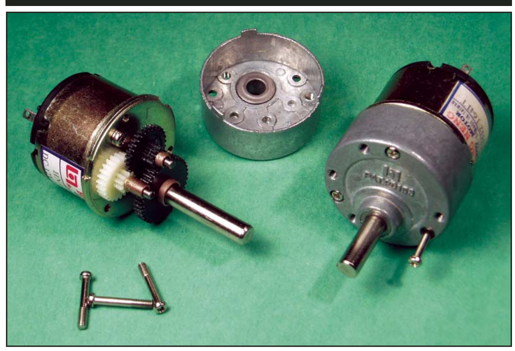


Figure 4. Removing gears to gain access to the gearbox mounting screws.

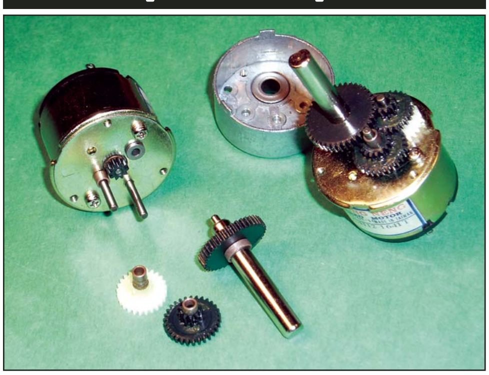


Figure 5. Removal of the gearbox mounting plate.

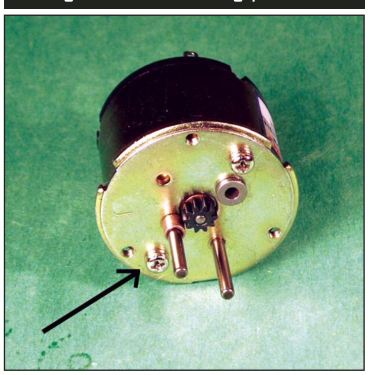
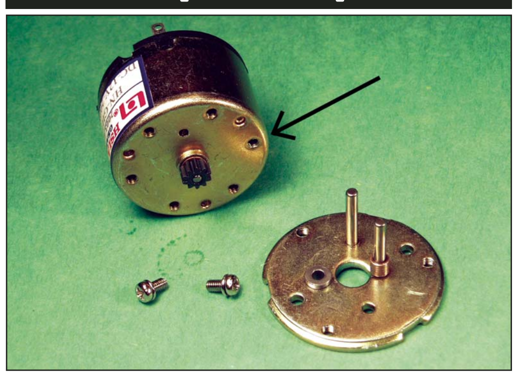


Figure 6. The 60-degree orientation mounting holes for the gearbox.



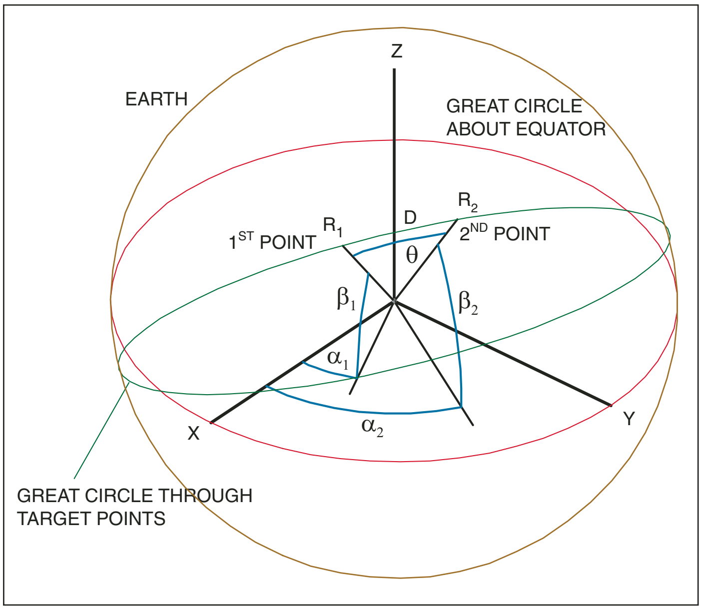


Figure 7. Sketch of a Great Circle passing through two points — R_1 and R_2 — on the surface of the Earth.

have a new motor with twice the torque as before.

Q. I have a Garmin E-Trex GPS receiver that I am thinking about using in my all-terrain robot to measure how it moves in my neighborhood. The GPS unit itself records how far it has gone, but I would like to know how this is calculated so that I can write a VB program to monitor the GPS unit to calculate how far my robot has traveled relative to various target points.

— Dan Kidwell

A. The easiest way to calculate how far you have moved between two sets of GPS (Global Positioning Satellites) coordinates is to calculate the distance using the sphere's Great Circle. A Great Circle is a circle where the edge of the circle passes through two points on the surface of a sphere, and the center of the circle is at the center of the sphere. The circle surrounding the equator is one example of a Great Circle.

When a robot is traveling along a straight line on the surface of the Earth, it is actually traveling along an arc path with respect to the center of the Earth. By geometry, we know that the total path length of an arc is the radius of the circle multiplied by the angle between the start and finish points of the arc. In the case of a sphere, the distance traveled between two points on the surface of the sphere will be the radius of the sphere multiplied by the angle between the start and finish points of the path. The equation for this basic relationship is shown below, where D is the traveled distance, R is the radius of the sphere, and θ is the angle between the start and finish points of the path. Remember, the angle here is measured in radians, not degrees.

$$D = R\theta$$

$$\textit{Angle (in Radians)} = \textit{Angle (in Degrees)}\,\frac{\pi}{180^\circ}$$

Figure 7 shows a simple sketch of the Earth with an X, Y, and Z axis coordinate system drawn at the center. Points R_1 and R_2 represent the start and finish points of the path. The distance traveled, D, is between these two points, and the angle between these two points is θ. The green circle in this sketch is the Great Circle that is created by these two points; α_1, α_2 represent the two GPS longitude angles for these two points, and β_1, β_2 represent the two GPS latitude angles for these two points.

Now it is time for a little analytical geometry (i.e., math). If the start and finish points — R_1 and R_2 — are defined as vectors with respect to the center of the Earth, they will lie in the same plane as the Great Circle. These vectors are defined below. The radius of the sphere is R.

$$\vec{R}_1 = R\left(\cos\beta_1\cos\alpha_1\hat{i} + \cos\beta_1\sin\alpha_1\hat{j} + \sin\beta_1\hat{k}\right)$$

$$\vec{R}_2 = R\left(\cos\beta_2\cos\alpha_2\hat{i} + \cos\beta_2\sin\alpha_2\hat{j} + \sin\beta_2\hat{k}\right)$$

The angle between these two vectors can be found by taking the dot product of the two vectors and dividing them by the product of their magnitudes. This is defined with the following equation. The next equation is the result of this derivation.

$$\cos\theta = \frac{\vec{R}_1 \cdot \vec{R}_2}{\left|\vec{R}_1\right|\left|\vec{R}_2\right|}$$

$$\cos\theta = \cos\beta_1\cos\alpha_1\cos\beta_2\cos\alpha_2 + \cos\beta_1\sin\alpha_1\cos\beta_2\sin\alpha_2 + \sin\beta_1\sin\beta_2$$

Thus, simplifying this equation and solving for the angle and multiplying it by the radius of the Great Circle, the distance between two points on the surface of the Earth is:

$$D = R \times \arccos\left(\cos\beta_1\cos\beta_2\cos(\alpha_1 - \alpha_2) + \sin\beta_1\sin\beta_2\right)$$

where, R is the radius of the Earth and the β and α angles represent latitude and longitude angles (respectively) for the start and finish points from a GPS unit.

One of the things to keep in mind when using this equation is precision of the math software you are using to make the cosine calculations. When the points are relatively close, the round off errors with the cosine function can result in large positional errors. Thus, the following equation is more commonly used. This formula is known as the Haversine formula.

$$D = 2R\arcsin\left(\sqrt{\sin^2\left(\frac{\beta_1 - \beta_2}{2}\right) + \cos\beta_1\cos\beta_2\sin^2\left(\frac{\alpha_1 - \alpha_2}{2}\right)}\right)$$

In the Seattle, WA area, the error between these two equations for a 0.001-minute of angular movement in distance depends on what is used to calculate the distance. For example, using Microsoft's Excel spreadsheet program, the error difference is only 0.015 inches which, for all practical purposes, is negligible. Now if I use my HP 48SX calculator, the error is now 48 inches for the same set of coordinates. With the calculator, the original equation

showed that there was no movement, i.e., the distance was zero inches, whereas the Haversine formula showed that the distance moved was 48 inches! This Haversine formula produces the same results with the calculator and Excel spreadsheet.

There are a lot of other factors that will affect the accuracy of the distance calculations, such as the actual radius of the Earth (it ranges from 6335.4 km to 6400.0 km), the accuracy of the GPS measurements (10 meters, even though three decimal places in the minute category would lead you to believe you have 2 m accuracy), and the elevation, to name a few.

Another thing to keep in mind when using these equations is that they assume there is a straight line movement between the two points. If your robot is changing directions from time to time, then you will need to calculate the distance moved on a periodic basis, and add the incremental moves up to obtain a total distance moved. Think about using it like an electronic odometer.

The equations shown here should help you get started with calculating distance movements with your GPS unit. On this month's download section at *SERVO Magazine*'s website (**www.servomagazine.com**), you can obtain a detailed explanation on how these equations are derived. Also, a search on the Internet has many web pages that discuss how to do this, and have Java-based calculators that make the same calculations. **SV**

```
// castling bonuses
B8 castleRates[]={-40,-35,-30,0,5};

//center weighting array to make pieces prefer
//the center of the board during the rating routine
B8 center[]={0,0,1,2,3,3,2,1,0,0};

//directions: orthogonal, diagonal, and left/right
from orthogonal for knight moves
B8 directions[]={-1,1,-10,10,-11,-9,11,9,10,-10,1,-
1};

//direction pointers for each piece (only really for
bishop rook and queen
B8 dirFrom[]={0,0,0,4,0,0};
B8 dirTo[]={0,0,0,8,4,8};

//Good moves from the current search are stored in
this array
//so we can recognize them while searching and make
sure they are tested first
```

LESSONS FROM THE LABORATORY

A bi-monthly column for kids!

LEGO Mindstorms NXT Software
Beyond the Common Palette

by James Isom

This month, we will continue our look at the educational version of the new NXT software. By the time this issue hits the newsstand, the NXT will be shipping and widely available. For more information on where to purchase a NXT Mindstorms robotics set, visit **www.legoeducation.com**

In the last article, we took a tour of the new programming environment and had a look at the basic functionality of the icons that live in the Common Palette. Continuing along where we left off, this time we'll complete the tour by looking at what lurks in the "Complete Palette."

Data hubs.

Data Hubs, Wires, and Plugs

Before we dive into the depths of the Complete Palette, I want to take you on a short detour through that which lurks below each programming block.

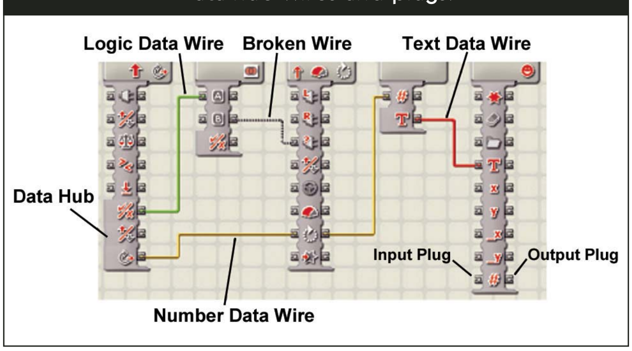
Below and slightly to the left of each block is what appears to be a partially exposed tab — clicking the tab will extend the "Data Hub."

Data such as text, numbers, or a bit of Boolean true/false logic can be passed around and used by programming blocks through the use of their data hub, wire, and plug structure. Data Hubs consist of one or two rows of plugs (the left are inputs, the right outputs) that can be connected together with data wires. "Data Wires" are color coded into one of four colors in accordance with their data type. Only certain types of wires will connect with certain plugs.

For example, you can't connect a text output plug to a number input plug. Doing so would result in a broken wire (gray) and cause your program to error when you try to download it to the NXT. Broken wires can be removed by simply selecting them and pressing the delete key.

Using connections between data hubs is useful for a variety of purposes like passing text from a "Text" block to a "Display" block or a number from a sensor's output plug to a "Math" block. There are hundreds of ways to use the hub, wire, and plug system and many of the most exciting features of the programming blocks in the Complete Palette rely on using it. Let's take a

Data hub wires and plugs.



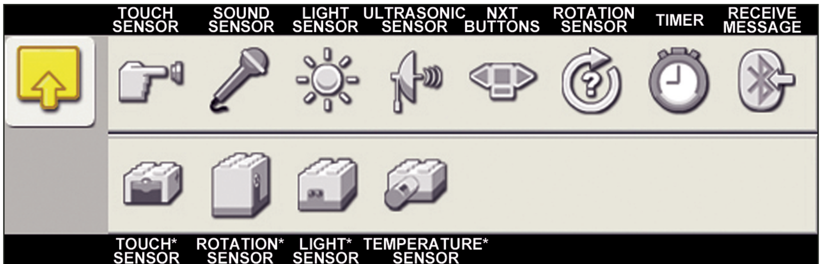
The Common Submenu.

look at the Complete Palette to see what it has to offer.

The Complete Palette

There are six main block categories in the complete palette, each representing a family of functions. Each expands into a submenu of blocks that can be used in your program.

The Action Submenu.



The Sensor Submenu.

Complete Palette.

The Common Menu

The blocks in the Common Menu are identical to those found in the Common Palette. With the exception of the Move and Record/Play blocks, all the other blocks in the Common Menu appear in other places in the Complete Palette. If you would like to know more about these basic functions, please read my article entitled "A Sneak Peak at the NXT Software" in the June '06 issue of *SERVO*.

The Action Menu

The aptly named "Action" menu holds programming blocks that do things or are otherwise outputs. These are responsible for the direct external behaviors of your robot like turning on a motor, playing a sound, or writing a picture to the NXT screen.

• *Motor* — Whereas the "Move" block allows you to set parameters for multiple motors at once, the motor command gives you control over just one. Acceleration can be governed through a control that sets whether the motor is ramped up slowly, started at full power, or ramped down. A Control Motor Power feature is also available that attempts to compensate for any resistance the motor encounters.

• *Send Message* — The Send Message block utilizes the NXT's Bluetooth wireless capability to send a message to another Bluetooth-enabled device such as your computer or another NXT.

Notice that the last two blocks in the graphic have an asterisk following their name. They represent the motor and the lamp from the legacy RCX system. Motors, lights, and sensors from the RCX system can be used on the NXT through the use of the conversion cables that come with the LEGO Education Base Set.

The Sensor Menu

The Sensor Menu has two rows of programming blocks. The top row holds all the new sensor blocks and the lower row holds the legacy RCX sensors (marked by an asterisk). The functionality of the sensor blocks is predictable, after each is selected and placed, the configuration panel changes to show the available options for that sensor. The first four sensors in the top row — Touch, Sound, Light, and Ultrasonic — appear in the Common Palette under the Wait For menu. The next four blocks are unique to this menu; a brief description of each follows:

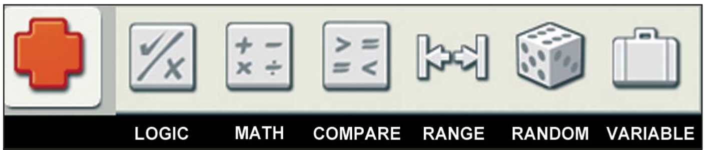
• *NXT Buttons* — The four buttons on the face of the NXT are now programmable and can serve as programmable controls or triggers for your programs. Two NXTs, a Bluetooth connection, and a little programming and you could make a remote control for your robot.

• *Rotation Sensor* — We've seen rotations before associated with the Move block from the Common Palette, but here the rotations are on their own, detached from a powered motor. Very useful when you want to monitor rotations, but don't want to power up the motor to do it.

• *Timer* — There are three of them to monitor or reset. They all start ticking away when your program is started. See the sample program included with this article for an idea on how to use one.

• *Receive Message* — The sibling of the Send Message block, these two can be used together to enable two or more NXTs to pass data back and forth.

All the blocks in the Sensor Menu can be used as triggers to activate or stop portions of your program. Triggers can be as simple as waiting for a touch sensor to be pressed or as complex as waiting from a specific value to be received by another NXT.



The Data Submenu.

The Flow Submenu.

The Flow Menu

The Flow Menu blocks control how a program progresses from start to finish. Wait, Loop, and Switch we are already familiar with, but the last in the menu is new.

• *Stop* — The name says it all. The Stop block brings your program to a halt, stopping all motors, sounds, etc. It's the one block that doesn't have a configuration panel so there aren't any options to set.

The Data Menu

The blocks in the Data Menu are used to manipulate data as it courses throughout your program.

• *Logic* — This is an important one so I am going to take a little time to explain it. Sometimes in a program, it's important to wait for a couple of things to happen before moving on to the next thing. A simple example of this might be if our robot was looking to pick up an object of a certain color, what questions should it ask before closing the gripper around the object? It should probably ask something like the following:

– Is the object the right color?
– Am I close enough to the object to grab it?

Let's say the robot was using a light sensor to monitor the object color and a touch sensor to detect if it was close enough to the object to pick it up. If either of the programmed events happen as programmed, the sensor blocks will send a "true" statement to the Logic block. If not, it will continue to send a "false" statement. The Logic block waits for the two questions to be true before sending out a true statement itself that will trigger the next part of the program. This is called an AND operation in the sense that the Logic Block is waiting for a true answer from both question #1 *and* question #2.

The Logic block uses something called a truth table to decide whether to send out a true statement. The truth table lists all the possible outcomes for a question.

The AND operation is just one of several logical operations a robot can use to evaluate inputs. OR sends out a true signal if either Input A or Input B is true, while XOR simply waits for a mismatch in the inputs.

The last type of operation in the Logic block list is NOT which basically receives an input and sends out the opposite, as in TRUE IN = FALSE OUT, FALSE IN = TRUE OUT. Once again the Help file has a pretty good explanation of all this or you could always Google "truth table" or "Boolean Logic" for more in-depth information.

• *Math* — Need to convert a number of rotations from your motors to the distance your robot has traveled so that you can display it on the screen? The Math block is your friend allowing you to add, subtract, multiply, or divide any two input numbers.

• *Compare* — This block allows you to compare two inputs using *less than*, *greater than*, or *equal to*. If your condition is met, the output signal can be used to trigger other parts of your program.

• *Range* — Is used to monitor whether an input is inside or outside a set of numbers.

• *Random* — Is used to generate a random number for your program. You can control the range from which the number is generated by setting a minimum and maximum value. Want a number between 1 and 10, 1 to 100, or 14 to 28? Random is where you get it from.

• *Variable* — A Variable is like a Container in ROBOLAB. It's a place to store a value that can be read or changed by

AND truth table.

"AND" OPERATION		
INPUT A (QUESTION 1)	INPUT B (QUESTION 2)	OUTPUT
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

XOR truth table.

"XOR" OPERATION		
INPUT A (QUESTION 1)	INPUT B (QUESTION 2)	OUTPUT
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE

different blocks throughout your program.

The Advanced Menu

• *Text* — With this block, you can combine text coming from another block with one or two lines from this block. This becomes especially handy when you want to display a bit of text from a sensor reading on the NXT screen followed by an explanation. For example, the number of centimeters traveled or the number of times the robot has hit an object.

• *Number to Text* — To a computer, numbers are not text and only text can be displayed on screen. So, if we continue with our last example above where we were displaying the number of times the robot hit an object, we would need to take the "number of times" from a variable and convert that to text before writing it to the screen with the Display block. The Number to Text block does that conversion for us.

• *Keep Alive* — This block acts like a little boost of coffee for your robot. It consists of a timer that keeps your robot from going into sleep mode until it has expired. The time value is fed to it by data wire.

• *File Access* — Like the data logging features on the RCX, this block allows you to read and write text files onto your NXT, which is helpful for keeping high scores for a game or data from a long-term experiment.

The Advanced Submenu.

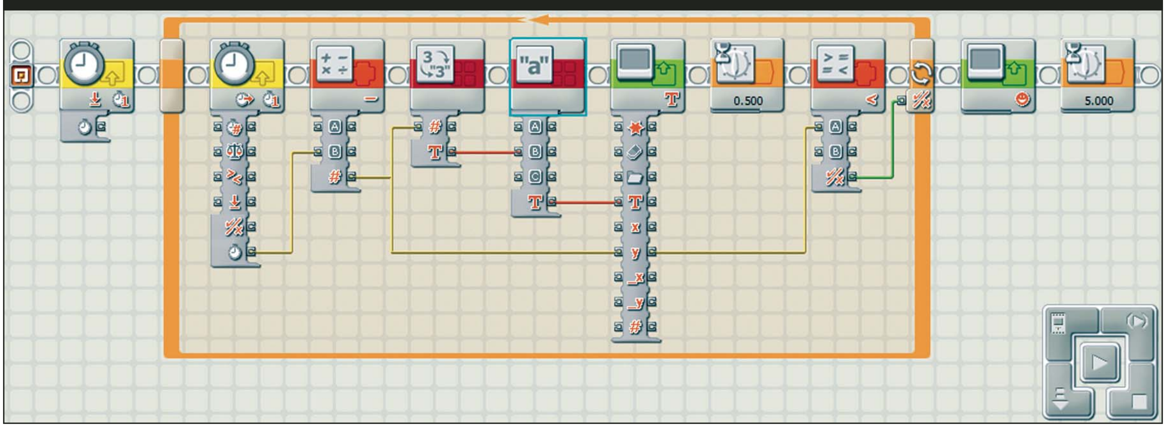
• *Calibrate* — This block allows you to set minimum or maximum values for an analog sensor. It takes two blocks to calibrate both values. This will be very handy for getting light sensor readings on the fly during a FIRST LEGO League Tournament.

• *Reset Motor* — The NXT servo motors have a default routine that keeps them accurate. Depending on the project, this accuracy might not always be a desired feature. The Reset Motor block allows you to turn off the auto correction routine for a motor.

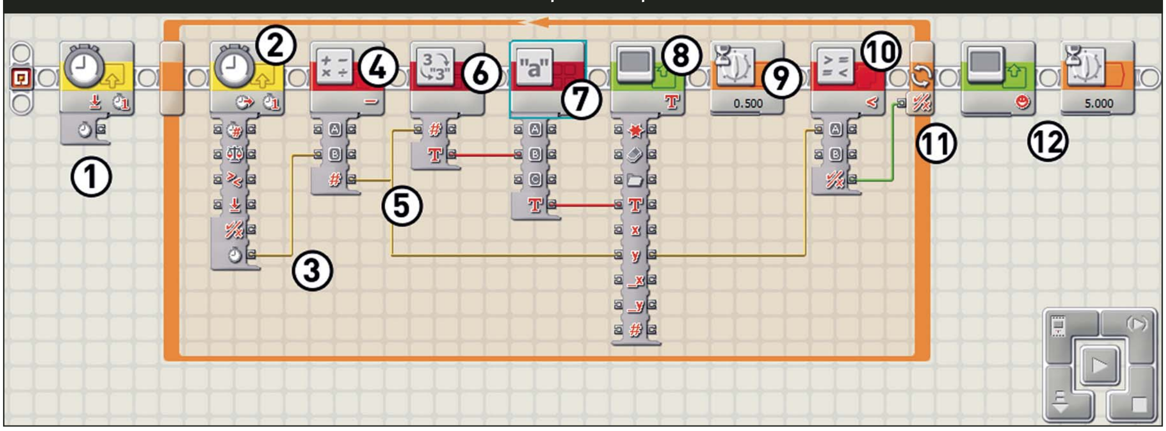
That just about does it for the Complete Palette. Let's have a look at a sample program using some of the new blocks. Can you guess what it does?

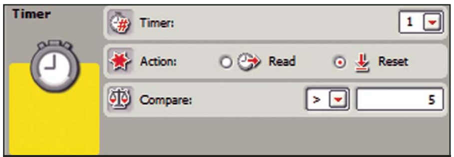
If you guessed that it displays text, you were right! This pointless little hack takes a string of text and scrolls it from the bottom to the top

Example 1.

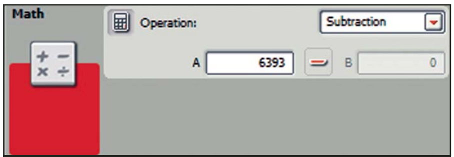


Example 1 steps.



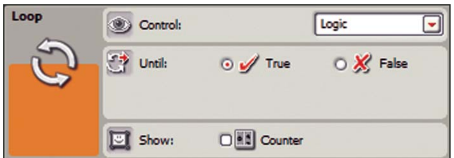


Example 1, Step 1.

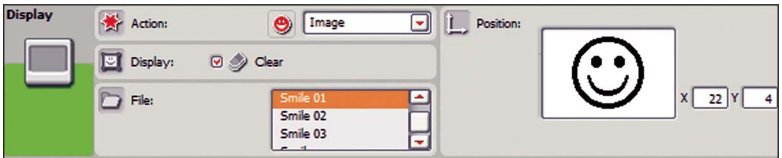


Example 1, Step 4.

Example 1, Step 7.



Example 1, Step 11.



Example 1, Step 12.

of the NXT's screen. In a more elegant form, it could be useful in future projects to display high scores from a robot game or roll the credits on a NXT Animated movie. This is how to put it together.

1. Take the Timer block from the Sensor Menu to reset Timer #1.

2. Place another Timer block inside a Loop and set it to read Timer #1.

3. Connect a data wire from the Timer Value output plug to the Math block's B input plug. The data wire sends the timer value out in milliseconds.

4. Use the Math block to subtract the value of input plug B from number 6393 placed in A. The screen is 64 pixels high. By taking the timer value and subtracting it, we get a value we can use to change the y-axis of the display that gives us our scroll effect.

AUTHOR BIO

James Isom is a part-time robotics teacher and generalall-around geek. He has taught robotics to children and teachers in the US and abroad. His website with additional goodies (including the MLCAD file of this robot) can be found at **www.therobot icslab.com** He can be reached at james@megagiant.com

5. Connect a data wire from the Result plug of the Math block to the y-axis input plug of the Display block. If you want to monitor the y-axis coordinates for troubleshooting purposes, make another connection from the Result output plug of the Math block to the number input plug of the next block.

6. Number to Text does its job by taking the input number and converting it to text before passing it out the Text output port.

7. The Text block combines anything on the three lines of text together on one horizontal line. It is important to remember to leave spaces at the end of your lines if you want to avoid mashing all your text into one big word. In A, the text "Hello World! " followed by a space has been input. The text in B is provided dynamically from the data wire coming from the Number to Text block. C is left blank and will not appear. The resulting text should look something like — "Hello World! 53" with the numbers changing as it scrolls up the screen.

8. The text is sent to the Display block one time as the program loops. Configure the Action Panel for Text. In Display, the Clear box should be checked. If it is not, each line of the screen will be filled with text like LEGO bricks from top to bottom — try it! Remove all the text from the Text field (i.e., Mindstorms NXT). Lastly, in the Position panel, set the X-coordinate to 5 and the Y-coordinate to 0. The Line menu should read 8. Connect a data wire from the Y-coordinate output plug to the A input plug of the Compare block.

9. Wait 0.5 seconds. A simple Wait block using time gives us a nice little interval. Play with the number and see what happens. (Warning: It doesn't take much to change things.)

10. The Compare block should be receiving data dynamically from the Display block into the A field. B should be set to 1 or some small number. The operation menu should read Less than. This block is waiting for the coordinates to drop below 1 before it sends out a true statement to end the loop.

11. The Loop does its thing until it receives a true signal from the Compare block.

12. Once the loop is stopped, I have it display a smiley face image for five seconds as an indicator that it has finished.

Our tour of the Complete Palette is now ... er ... well ... complete! Next time, we'll start a new project with the NXT. Until then, have fun! **SV**

Rubberbands and BALING WIRE

by Jack Buffington

A Real Looker

How to Let Your Robot See

There are many types of sensors that people can add to their hobby robot projects, but one type of sensor that you almost never see is an image sensor. For humans, sight is our primary method of getting information about our world. Wouldn't it be great to allow our robots to perceive the world in the same way that we do? This month's column will show you how you can add an image sensor to your robot so that it can learn much more about the environment that it is in.

The image sensor that will be used is not what immediately comes to mind when you think of an image sensor. This column will be using the Taos TSL3301. It is a linear array of 102 pixels. This chip comes in a clear eight-pin DIP package, which makes it handy for those of us who like to prototype on breadboards or on perfboard. This chip can divide its array into three 34-pixel sections. Each section can have a separate gain and offset values though this column will set all three sections to the same settings. This chip can run off of a single five-volt supply and has a completely digital interface. This is quite handy when you are using a low-end microcontroller that doesn't have an analog-to-digital converter.

The TSL3301 has one of the smallest number of pixels in the series of chips that Taos produces, but this small number fits well with small embedded processors that have limited amounts of RAM to use. Despite the RAM limitation, generally your robot is going to have all the time that it needs to process the information that it receives so the actual pattern recognition tasks shouldn't be much of a limiting factor.

Let's look at the pinout for the TSL3301. As you can see, it only has three pins that you will be using to communicate with this chip. This makes it really easy to interface with your microcontroller. The interface is a strange mishmash of the RS232 protocol and SPI.

The data lines communicate using one start bit, eight bits of data, and one stop bit as RS232 does, but this data can come and go at almost any baud rate because you are also providing a clock signal. You will need a pretty fast processor to hit its speed limit, which is a clock of 10 MHz.

The other quirk about this chip is that it has no internal clock to drive its functionality so the clock that you provide for the serial communications is also what drives its internal functionality. Because of this, sometimes you will need to send a few extra clock pulses to the chip so that it can finish doing things internally.

Before going further ahead into how the chip operates, let's back up and look at how to get an image projected onto the pixel array in the first place. Working with optics can be an involved process if you are trying to achieve a high quality image. Fortunately for us, having a low-quality image is more than sufficient for our purposes since we only have 102 pixels to capture the image anyway.

A single, double convex lens was used to project the image onto the chip. The lens was part number NT32-019, purchased through Edmund Industrial Optics. This lens is 9 mm in diameter and has a 9 mm focal length. Because of the short focal length, this

Figure 1. The pinout for the TSL3301.

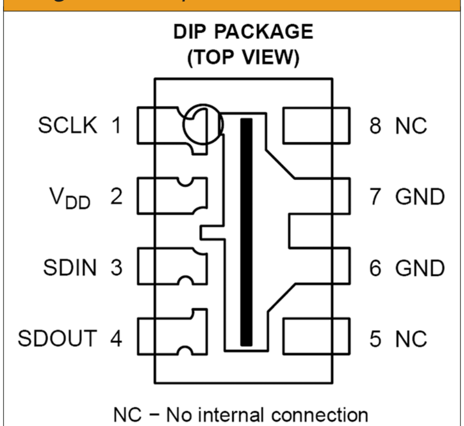


Figure 2. A side view of the image sensor assembly.

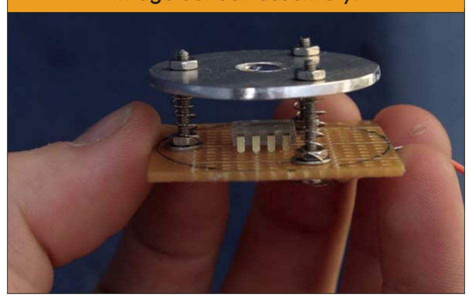
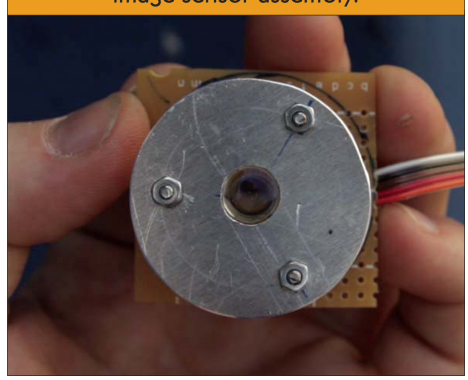


Figure 3. A top view of the image sensor assembly.



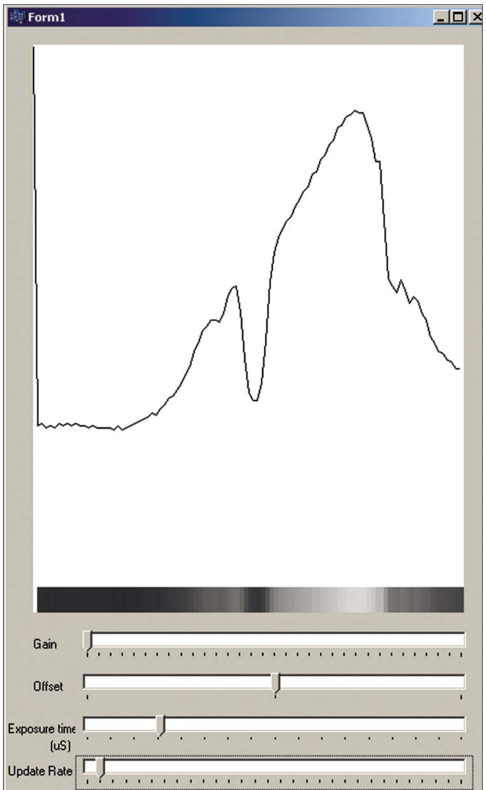


Figure 4. This program is on *SERVO*'s website and allows you to view the images from the TSL3301.

lens allows for a wide field of view. This can give you a good overview of the room that your robot is in, but won't allow you to see detail.

This lens was mounted, as shown in Figures 2 and 3. A piece of aluminum was cut into a circle and a hole was drilled in its center that was just slightly bigger than the lens diameter. Next, three holes were drilled into the perimeter of the aluminum piece that allow 1-72 bolts to pass through. Matching holes were drilled into a prototyping circuit board that the image sensor was mounted to. Then 1-72 bolts were put through the holes in the prototyping board and nuts were put onto the other side to keep them mounted firmly in place.

Small springs were made to go around the bolts. These springs keep the aluminum piece and lens away from the sensor. You can make a spring by wrapping piano wire around a drill bit or any other round piece of metal. Next, the lens is mounted to the aluminum piece by first laying the aluminum piece flat onto a table and placing the lens inside of its hole. Now take some super glue and put three drops of it around the lens on the aluminum. Make sure that no super glue touches the lens at this point.

Take a toothpick and carefully drag the drops over to the edge of the lens. Let this sit for a few minutes and your lens will be firmly bonded to the aluminum. Make sure that the super glue is fully dried or else you risk getting some onto the lens with your fingers when you pick it up. This type of mount is a little more involved to make than others, but allows for higher precision focusing due to the high number of threads per inch in the bolts.

Slide the aluminum disk over the three bolts and thread some nuts onto the bolts. These nuts won't be tightened but instead will allow you to adjust the distance of the lens from the image sensor. When you find the place that is in focus, put a little locktite onto the nuts to keep the lens in place.

Okay, you can now project an image onto your sensor, so let's go back to how to talk to this chip. This chip is really easy to communicate with. It does, however, require that you write your own bit-banged receive and send routines because of its quirky interface. The TSL3301 chip has three communication lines. These are called: SCK, which is the clock line; SDIN, which is the line that the chip receives data on; and SDOUT, which is its transmit line. SDOUT and SCK will be used to transmit.

Do the following to send a byte to the LTC3301:

- Drive the SDOUT line low.

- Pulse the SCK line by driving it high and then low again. If you have a fast processor, be mindful of the maximum clock rate of 10 MHz.

- Create a loop that repeats eight times and does the following:
  - – Look at the least significant bit in the byte that will be sent and set the SDOUT line to match.
  - – Pulse the SCK line.
  - – Shift the byte that is being output one bit to the right.

- Drive SDOUT high.

- Pulse SCK.

There is some source code that runs on a PIC16F873 processor that's available on the *SERVO* website (**www.servomagazine.com**) that you can reference if you are having trouble with something that you see in this month's column.

To receive a byte from the TDL3301, you need to do the following:

- Pulse the SCK line once to skip over the start bit.

- Clear a register that will hold the received byte. We'll call this DATA.

- Now create a loop that does the following eight times:
  - – Shift DATA one bit to the right.
  - – If SDIN is high, then it will set the highest bit of DATA.
  - – Pulse SCK.

- Finally, pulse SCK once to skip over the stop bit.

The TSL3301 chip needs to be initialized when you first power it up. Here is the routine that you follow to make it happy so that you can start sending it commands:

- Drive the SCK line low.
- Drive the SDIN line low.
- Pulse the SCK line 30 times.
- Drive the SDIN line high.
- Pulse the SCK line 10 times.
- Send 0x1B to the chip.
- Pulse the clock five times.
- Send 0x5F to the chip.
- Send 0x00 to the chip.

Before you start reading the data from the chip, you may want to change the gain and offset values. Gain adjusts the scaling of the values that are read. Increasing gain can add noise to the image but may be necessary if you are taking hundreds of images per second. The gain variable can be anything from 0 to 31. Offset adds or subtracts a fixed value from each pixel. It is an eight-bit sign magnitude variable so it can represent any value from –128 to 127.

To adjust your gains and offsets,

you will need to write to a few registers. There are three gain and three offset registers that correspond to the different 34-pixel sections of the array. To write to a register, first you will send its address and then the value that you want to write to it. The addresses for the offset registers are 0x40, 0x42, and 0x44. The addresses for the gain registers are 0x41, 0x43, and 0x45.

Now you are ready to capture your image. To capture an image, you will need to do the following:

- Send 0x80 to the chip to start capturing the image.
- Pulse SCK 22 times.
- Delay for the amount of time necessary to capture the image. This would be equivalent to how long the shutter would be open in a real camera. Shutter times of one microsecond to 255 microseconds make for a pretty good range that can see in bright sunlight and in candlelight.
- Send 0x10 to the chip to stop capturing the image.
- Pulse SCK five times.
- Send 0x02 to start reading the pixels from the chip.
- Pulse SCK repeatedly until you see a start bit (low SDOUT).
- For all 102 pixels, receive a byte.

Wow! There were a lot of things that you needed to set up, but once you have all of the routines that were described here written, you can start to have some fun with this chip. One thing that you might like to do with this sensor is to see in color. This sensor simply responds to the amount of light that strikes it, so if you want a color image, then you will need to use filters to read red, green, and blue images. You can then combine these to make a full-color image.

Buying professional optical filters can be expensive. A cheap way to get around that problem is to go to a local store that sells or rents motion picture, stage lighting, or maybe photography equipment. You can often find sample booklets of filters that are used to color lights. The samples are far too small to put over a light but are more than big enough to put over your robot's tiny lens. The nice thing is that these filter booklets have graphs of the colors that they allow to pass through so filter selection is easy. Making a filter wheel that rotates in front of your sensor would allow you to capture color images.

Something that you should be aware of is that if your robot is in a room with fluorescent lights, then your images will vary a lot in brightness due to the flickering of the fluorescent bulbs. You might want to put a dark filter over the sensor and increase your exposure time to a full cycle of the bulb's flicker rate; 8.3 milliseconds should work for fluorescents with older ballasts. Newer electronic ballasts might not create this flicker problem.

If you want an image that you can display on a computer, you could mount the sensor and lens onto a hobby servo and slowly sweep it around the room. The software that is provided on *SERVO*'s website allows you to see a graph of the brightness of each pixel and a grayscale version of what it is seeing, as well. It would be fairly simple to modify it into a program that progressively captured images and displayed them on successive columns or rows.

RESOURCES

Mouser Electronics
www.mouser.com
Sells the TSL3301 chip.

LEE Filters USA
www.leefiltersusa.com
Sells filters for the motion picture industry.

Edmund Optics
www.edmundoptics.com/US/
Sells lenses.

Custom Computer Services, Inc.
www.ccsinfo.com
Sells the C compiler used for the PIC code on *SERVO*'s website.

Borland
www.borland.com/us
Sells the C++ compiler used for the PC code on *SERVO*'s website.

Visual input is not something that you commonly see in hobby robotics though it isn't terribly difficult or expensive to integrate into your projects. There are endless possibilities of things that you can do with robotic vision. You could track moving objects. You could do optical range finding. You could locate objects of a certain color or determine the motion of something without any physical contact. What could you do with a sensor like this? **SV**

EVENTS CALENDAR

Send updates, new listings, corrections, complaints, and suggestions to: **steve@ncc.com** or FAX **972-404-0269**

Robots are probably not the first thing that comes to mind when you think of South Africa. An organization called the National Youth Development Trust (NYDT) is trying to change that with the creation of the Africa Cup Robotics Competition. Their goal is to make South African students, teachers, and robot enthusiasts globally competitive. They want their students to be able to compete with and against students from any other country in robotics competitions and other science-based events.

The Africa Cup is designed to promote participation online and in person throughout the year, culminating in an event scheduled to coincide with the African Youth Games, April 1-8, 2007 in Pretoria. The students will work in teams to prepare for events including obstacle course races, wall climbing, robot sumo, and robot soccer.

The organizers seem to have done their research. They have created event categories for pre-manufactured robots, as well as custom-designed robots. Beginners will be allowed to create remote-controlled robots, while more advanced builders will be able to create autonomous robots. Five levels of participation have been defined: Junior School, Middle School, Senior School, Professional, and Special Outreach (for students with disabilities).

The Africa Cup is just one of several science-based, academic challenges being developed in Africa. Others include model rocketry and aeronautics contests. I'm looking forward to seeing the results of their first robot competition. If you'd like more information on this event, visit the NYDT Robotics Program web page at **www.nydt.org/home.asp?pid=760**

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL, as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: **http://robots.net/rcfaq.html**

— R. Steven Rainwater

August

2-6 **AUVS International Undersea Robotics Competition**
US Navy TRANSDEC, San Diego, CA
Autonomous underwater robots must complete a course with various requirements that change each year.
www.auvsi.org/competitions/water.cfm

10-13 **Robot Fighting League National**
Minneapolis, MN
Four days of watching over 100 radio-controlled vehicles destroy each other.
www.botleague.com

20 **RoboCountry**
Takamtsu City, Kagawa, Japan
Described on the website as *humanoid robot combat* presented by the Kagawa Humanoid Robot Society.
www6.ocn.ne.jp/~robotics

September

1-4 **DragonCon Robot Battles**
Atlanta, GA
Radio-controlled vehicles destroy each other at a famous science fiction convention.
www.dragoncon.org

9 **SWARC Texas Cup**
Mike's Hobby Shop, Carrolton, TX
Radio-controlled vehicles destroy each other Texas-style.
www.robotrebellion.net

16-17 **RoboCup Junior Australia**
University of NSW, NSW, Australia
There are over 600 RoboCup Junior teams in Australia. Regionals narrow this down to about 200 teams who will compete at the University of NSW to see who's the best at building LEGO-based autonomous soccer robots.
www.robocupjunior.org.au

30 **Robothon**
Center House, Seattle Center, Seattle, WA
Events continue on October 1 for two full days of robot contests that include line-following, line-maze, Robo-Magellan, walker races, mini sumo, and 3 kg sumo.
www.robothon.org

ROBOTS.NET

October

14 **Robot-Liga**
Kaiserlauter, Germany
Includes mini sumo, line search, labyrinth, master labyrinth, robot volley, and robot ball.
www.robotliga.de

20 **Elevator:2010 Climber Competition**
Las Cruces, NM
Autonomous climber robot must ascend a 60 meter scale model of a space elevator using power from a 10 kW Xenon search light at the base.
www.elevator2010.org/site/competition.html

27-29 **Critter Crunch**
Four Points Sheraton Hotel, Denver, CO
In conjunction with MileHiCon. See robot combat by inventers of robot combat competitions.
www.milehicon.org

November

18 **DPRG RoboRama**
The Science Place, Dallas, TX
Events include Quick-Trip, line-following, wall-following, T-Time, and Can-Can.
www.dprg.org/competitions

24-25 **Hawaii Underwater Robot Challenge**
Seafloor Mapping Lab, University of Hawaii, Manoa, HI
ROVs built by university and high-school students compete. This event is part of the MATE (Marine Advanced Technology Education) series of contests.
www.mpcfaculty.net/jill_zande/HURC_contest.htm

Robotics Showcase

NEW PRODUCTS

CONTROLLERS

Low-Cost Audio Repeater Provides Synchronized Device Control

Eletech Electronics announces the immediate availability of its QuikWave model EM31A-X audio repeater with extended device control. EM31A-X is a Flash card based MP3 player with a built-in power relay. Through this relay, external devices such as solenoids, motors, and lights can be turned on and off at any time in perfect synchronization with the audio playback. One good example is a talking robot with synchronized mouth movement.

A single contact closure input is provided for triggering a pre-programmed animation cycle of up to 8.5 minutes. Within this period, the audio and the relay can be independently activated multiple times in any order at any time. Minimum timing resolution is 1/8 second, which is far superior than what other products offer (typically 1/4 second or less.)

Through the user friendly Teach-n-Learn technology, the animation sequence can be easily programmed into the system by pressing some buttons on the unit. Just manually perform the animation sequence one time for the system to learn. No computer or software is required to program or operate the unit.

EM31A-X provides stereo, CD quality line output to be amplified by external power amplifiers. Housed in a rugged metal enclosure measuring 6.6" x 3.1" x 1.5", it is built to provide years of reliable service with no required maintenance. Typical applications include museum exhibits, animated point-of-purchase displays, haunted house event control, etc. EM-31AX is now in full production with a list price of $249.

For further information, please contact:

Eletech Electronics Website: **www.eletech.com**

New Motion Controller

Trust Automation has introduced the innovative TA600 four-axis stand-alone Motion Controller for use with brushed, brushless, and stepper motor drives. The TA600 controller optimizes the performance and reliability of motion systems through the use of dual processors. For application program execution, host communication, and general I/O controls, a high speed microcontroller is used. For the motion-specific tasks, a DSP processor is used. This allows each processor to operate in its area of greatest reliability and highest performance.

Fully C language programmable, the default TA600 firmware uses an application proven, three letter command set, user programmable macros, programmable Enable, Fault, Home, and Limit levels. It also features an integrated Emergency Stop circuit for active or passive control of the complete system's safety features. The TA600 controller is ideal for: Gantry robots, pick-and-place, assembly, inspection, automation, laser and water cutting, and medical applications such as surgical robots.

The TA600 controller features: Point-to-point, Trapezoidal, S-curve, and Custom profiling; Linear Interpolation; Position Velocity Time profiling; Electronic Gearing; and Analog Feedback profiling, as well as Fast Event capturing inputs for the greatest degree of control possible. Optional dual Digital-to-Analog Converters (DACs, two per axis) provide for sinusoidal commutation of motors resulting in very smooth motion, especially when used with high performance brushless linear motors. Additionally, sinusoidal commutation of stepper motors results in performance similar to brushless rotary motors.

The one to four axis TA600 controller is able to integrate brush and brushless servo drives, and stepper drives for conventional and linear motors. For fast linear motor axes that require fast settling times, a very fast 50 μ sec servo update rate is incorporated. Three PID with feedforward tuning filters for each axis make standing, moving, and stopping stability easy to achieve. Dual bi-quad filters for each axis make taming bad system harmonics possible. Feedback from incremental encoders, Hall magnetic sensors, and/or 14 bit analog feedback from a single or dual loop system is handled seamlessly.

Compact, measuring just 1.5 in. wide (38.1 mm) x 8.1 in. high (205.7 mm) x 7.4 in. deep (188.0 mm), and weighing just 2 lbs (0.9 kg), the TA600 controller is designed for use with a 24 to 28 VDC supply at 0.5 to 6 amps (fused) and panel mounting.

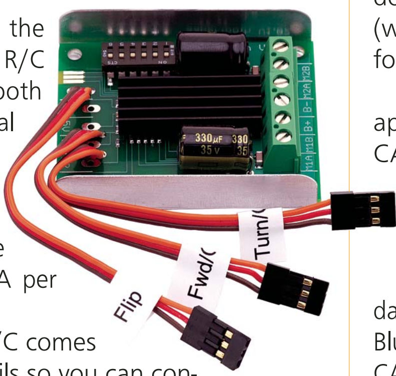
For further information, please contact:

Trust Automation

205 Suburban Rd.
San Luis Obispo, CA 93401
Tel: **805•544•0761** Fax: **805•544•4621**
Email: **info@trustautomation.com**
Website: **www.trustautomation.com**

MOTOR CONTROLLERS

The Sabertooth 2X10 R/C



Designed specially for the combat robot and R/C vehicle crowd, the Sabertooth 2X10 R/C is the latest dual 10A motor controller from Dimension Engineering. It accepts battery voltages from 6 to 24V and will handle peak currents as high as 15A per motor.

The Sabertooth 2X10 R/C comes with presoldered servo pigtails so you can connect it directly to your radio receiver. The product's options are set with DIP switches and include exponential control, autocalibration, safety timeout, and mixed (tank style) steering mode. A selectable lithium mode protects expensive LiPo batteries from damage due to overdischarge.

The invert/flip mode — used in combat tournaments — is unique because you can choose to toggle it with an R/C channel or a logic level signal.

As with their other motor drivers, Dimension Engineering's custom designed synchronous regenerative H-bridge topology returns the motor's stored inductive energy to the battery in every switching cycle. This technique results in your motors running cooler and extends battery life by 20-50%, depending on the motors used. It also provides more responsive control — allowing you to make instant stops and reverses.

Heatsinks come preinstalled and the unit has electronic thermal and overcurrent protection — making it a durable investment.

For further information, please contact:

Dimension Engineering	Website: **www.dimensionengineering.com**

SOFTWARE

CASL 4.3 Available

Wagware Systems, Inc., and Brainyware, LLC announce the immediate availability of CASL 4.3. CASL (Compact Application Solution Language) is a development environment which allows developers to quickly create applications for PalmOS®, PocketPC/Windows Mobile, and Windows platforms, using a singe code base. CASL also allows extending applications to take advantage of device-specific features using C Code (on the PalmOS) or Dynamic Link Libraries (DLLs) on PocketPC/Windows platforms. The Freeware version (available for immediate download) supports Serial (COM) port communications (wired or via Bluetooth), making CASL an ideal language for robotics or other electronic projects.

There are no run-time royalties to distribute CASL applications to end-users of CASL applications. This makes CASL particularly attractive to corporate, shareware, commercial, and freeware authors. CASL strikes an optimum balance of core capabilities, extendibility, and features on all three supported platforms.

Key features include hi-res graphic support, indexed database files for fast random access, communication with BlueTooth serial devices using the BlueTooth library (included), CASLpro (Palm) Events Library that allows detection of: the Five-Way navigator (DPad), Hard Power Off/On, Soft Power Off/On, Launch Handler (to prevent or allow application exit), and Pen Up/Down/Move, grid object with columns, which can be populated with label, textbox, button, checkbox, and dropdown objects. Each object may be bound to Database fields (or arrays) without writing any code, conduit for synchronizing PDA with the desktop computer (Hotsync® on PalmOS® and ActiveSync 4.x support on PocketPC / Windows Mobile), network socket access (HTTP, Telnet or TCP) for connecting to Wi-Fi, Ethernet, etc; and serial port access.

For commercial/corporate developers, the price is $299 for new users; $199 for upgrades (CASL 4.1 or lower license required for upgrade). CASL is free for non-commercial use.

For further information, please contact:

Wagware Systems, Inc.	Email: **sales@caslsoft.com** Website: **www.caslsoft.com**

COMBAT ZONE

PARTICIPATION

Robotic Safety

● by Russ Barrow, SWARC President

As robotic sports continue to grow, safety for the competitors and audience must be upheld. The modern robot has gained significantly in speed, agility, and power, and with this power comes responsibility. As a builder and organizer, the future of this sport depends on people operating robots safely and in a manner respectful of others. As someone who has been in the sport for several years, I have the responsibility of sharing and maintaining a safe environment both at home and at Southwestern Association of Robotic Combat (SWARC) events and meetings. This is the role of the Safety Coordinator.

Robotic combat is certainly not the only robotic sport that requires careful consideration of safety, but whenever electricity and mechanical systems come together for the first time or in competition, perils abound. From my work with robotics, I can share some lessons that I have learned, sometimes ones I have learned the hard way.

Building Environment

It is difficult to keep the area around a robot construction clean, but at the very least, be cognizant of debris that is inherently dangerous. Metal work produces tiny metal shards that make for painful splinters and slippery surfaces. Carbon fiber is especially hazardous to work with since the dust created by cutting or sanding can be very damaging to your lungs if inhaled. Inhalation hazards also include paints, welding fumes, glues, carbon monoxide, and cleaning agents. Always work in a well ventilated environment whenever these are used. Use gloves and eye protection when working with power tools, regardless of the size of the job or the power of the tool.

Testing

When the build is finished, the last thing you think about is what you will do if everything does not work. Shorts in wiring, bridged connectors, exposed components, and batteries can quickly burn off wire insulation and escalate the potential for a fire. For this reason, always keep a fire extinguisher near your test area.

Combat robotics adds an additional risk for first-time tests or stress testing in general. The adrenaline you feel during that first test is your body informing you of the risks you are taking. The best test environment would be an arena built for competing, but most people do not have easy access. Use common sense, and test in the order of least risk. This means placing the robot in such a way that the drive wheels will not move the robot, and lock-out all weapons to prevent movement.

First, test your RF communication (including failsafes), then power the robot. Test the drive and weapon operation. Drive and weapons both need to be tested at low power levels, whenever possible. Always be aware of the immediate path for the drive of the robot and the weapon.

Competing

Each event and sports competition has rules for safety inspection and competing. Most of these rules exist to protect the builder and the immediate public, and have likely evolved to accommodate the venue or class of robot. Read and understand the rules, and keep in mind the purpose of the rule, not just the language. The Event Organizer and Safety Coordinator will enforce these rules and always reserve the right to allow a robot to compete.

The robot safety inspection will typically consist of a visual inspection of the robot as it is unpowered. This may include a review of the power system and internal inspection of the robot. The Safety Coordinator may not be aware of all potential risks of the robot, so be sure to communicate any special or unknown functions.

Dimension and weight restrictions will also be inspected at this stage. The final test will consist of the robot being placed in the competition environment, which will test for controllability, function, and safe power-up/power-down.

When preparing for the match, be aware of the hazards around you. Follow the instructions of the Safety Coordinator, since they will advise on the safest method for loading and unloading into the competition environment. Finally, keep the robot powered down and immobilized between matches or when transporting. Remember, others are counting on you to be responsible and safe. SV

Big Combat Bots — Bargains or Bankruptcy?

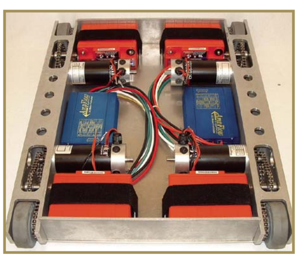
● by Kevin Berry

In the glory days of televised robot combat, there were dozens (or maybe hundreds) of bots in the 220 lb and up weight classes. At Robocide — an event held in January 2003 that drew many of the post-BattleBots competitors — there were 31 big bots registered. The 2003 Nationals registered 29. Robogames 2005 drew 32. But this year's Robogames (just at the time of this writing) has only 19. Most regional level events — like Battle Beach or Mechwars — have trouble rounding up a dozen or so these days.

Explaining the gradual decline in the number of big bots has been a matter of huge debate among builders and organizers. The explosion in the quantity of smaller bots (Robogames had 93 bots, 12 pounds and under, registered) may show that old time builders are shifting to smaller bots, and entry builders are opting for them, also.

Hardware cost, handling concerns, shipping costs, and fabrication time all are, of course, much higher when builders go for big bots. That, plus the sheer destructive power of these monsters, intimidates new builders. Press releases for mega events often brag that "some of the bigger bots cost upwards of $20,000" which further alienates uncertain new builders. In an attempt to understand the true cost of big bots, and possibly stimulate builders into restocking the ranks of Heavyweight and Superheavyweight fighting bots, I did some research into what it really takes to bring an economical, combat-worthy machine into being.

Like many areas of life involving do-it-yourself projects, answers from builders surveyed fell into a broad range of solutions. These have been lumped together into general categories: "Off-the-Shelf Kit Bots," "Partial Kit Bots," "High-End Home Builts," and "Basic Machines." This article will show potential builders that — surprisingly — it's not nearly as expensive as you might think to field a large, simple, combat-worthy, remote-controlled fighting bot. (Note: All parts lists and prices are done without radio systems, as comparing them is a whole different article's worth of material. Assume that any 75 MHz FM PCM system or higher will work for any bot in this article.)


BattleKit configured as a Superheavyweight.

Off-the-Shelf Kit Bots

Carlo Bertocchini — creator of the famous "BioHazard" — markets a series of platform kits that are highly regarded by the community. Under the banner of "BattleKits,"

Bot	Parts Included	Price
BattleKits HW	Chassis, one AmpFlow two-channel controller, two Hawker PC680 batteries, two S28-400 motors	$2,372
BattleKits SHW	Chassis, two AmpFlow one-channel controllers, four Hawker PC545 batteries, four S28-400 motors	$3,905
Team Moon SHW	Chassis, one AmpFlow two-channel controller, two AmpFlow Intercooled batteries, four S28-150 motors	$2,930

Table 1. BattleKits Configurations (Note: Shipping may be up to $150).

Bot	Parts Included	Price
Robot MarketPlace Basic HW	Power Switch, two IFI 885 controllers, four NPC-41250 motors, wheels, wire	$1,021
Robot MarketPlace Advanced HW 2WD	Power Switch, two Vantec RDFR33 controllers, two NPC T-64 motors, wheels, wire	$1,209
Robot MarketPlace Advanced HW 4WD	Power Switch, two Vantec RDFR36E controllers, four NPC T-64 motors, wheels, wire	$1,860

Table 2. Robot MarketPlace Configurations (Note: Shipping may be up to $150).

Supplier	Parts List	Price
BattleKits	HW Chassis	$899
Surplus	Gearmotors	$160
New	Sealed Lead Acid seven amp hour batteries, two IFI 883 controllers	$500
	Total:	**$1,559**

Table 3. BattleKits Part Kit Configuration (Note: Shipping not included).

Carlo sells kits for Featherweights, Lightweights, Middleweights, and a Heavyweight (HW) platform that can be upgraded to a Superheavyweight (SHW) (**www.battlekits.com**). Using the online selection and pricing tool, I priced out the two recommended configurations. Billy Moon — leader of Team Moon Robotics (**www.team-moon.com**) and a noted big bot builder — created his ideal machine, which differed slightly. Results of all three pricing runs are shown in Table 1. Descriptions of the recommended HW and SHW configurations from the BattleKits website are:

"The heavyweight ships fully assembled (except for electrical connections) ... It is 29.90" long by 24.00" wide. It weighs ... 87 pounds with two S28-400 motors, two PC-680 batteries, and the AmpFlow speed controller.

You can configure this kit as a superheavyweight (340 pound class), with four S28-400 motors, four PC-545 batteries, and dual AmpFlow controllers. It weighs 122 pounds in this configuration leaving you over 200 pounds to use for armor and weapons."

The builder must supply armor and weapons to the platform, so an additional cost of "nothing" to "expensive" (depending on the builder's solution) will be added to the costs in Table 1. Realistically, a couple of hundred dollars would buy a lot of armor for a pushybot, while weapons can range from a few hundred to many thousands.

To this author's knowledge, no other manufacturer currently offers combat robot platforms at this level of assembly. So, let's move to the next building level: part kit/part home built bots.

Partial Kit Bots

Another premier source — The Robot MarketPlace (**www.robotcombat.com**) — sells drive components as a kit, although not the chassis. Their Basic and Advanced Robot Starter Kits are a good source for builders who are willing to take on the mechanical design, but are a bit unsure about matching all the active components. The Robot MarketPlace is run by another veteran builder, Jim Smentowski, builder of famous heavyweight Nightmare. Info from his site about the Basic kit:

"You could ... choose four motors and have a great setup for a four-wheel-drive heavyweight (220 lb) robot. With this complete package, you will have all the main components ready to go.* Simply hook everything together on your own platform or chassis, and you'll be driving your bot around soon! For a basic robot, all you'll need to provide on your own, beyond this package, are the following items (all of which you could pick up at your local hardware store and electronics store):

- Batteries (we would include with this package, but there are too many choices)
- A metal or wood baseplate and/or frame
- Fasteners (nuts, bolts, and wire connectors)
- A weapon (optional)"

The Advanced kit adds:

"These parts are selected to work well for a ... two wheel drive heavyweight (220 lb) robot. Add two more motors and upgrade your speed controller to go with four wheel drive."

So, let's look at the cost of the Basic and Advanced Packages, in the various recommended configurations, shown in Table 2.

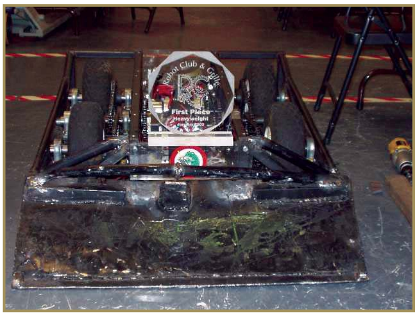
Of course, all these require adding batteries and some sort of chassis. Battery cost is around $100–$400, and chassis costs are variable — from free to expensive — depending on the material and fabrication costs. So the cost, less armor or weapons, of a functioning HW

platform is probably in the $1,200–$2,700 range. (Again, radio system is extra in all pricing comparisons.)

BattleKits suggested a partial kit/part home procured setup that is popular among builders. This configuration is shown in Table 3.

Head-to-head price comparisons are hard to make, since the amount of fabrication, material costs, and prices of surplus hardware varies radically with each builder's unique design for their bot.

High End Home Builts



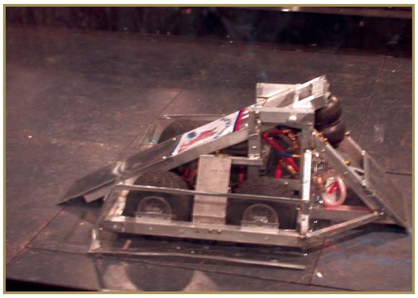
Heavyweight "Evelyn" is a great example of a simple, well-built bot.

Steven Kirk Nelson, from Team K.I.S.S. (**www.teamkiss.com**) supplied a components list and some fabrication details from his winning HW "Evelyn." His thoughts are:

"The Heavyweight Evelyn I was running in 2003-2004 was basically a super drive train with a lighter frame and a slightly higher gear reduction 12-to-1. It uses two three-inch S28-400 Magmotors and a Vantec 38E speed controller. For a Super, I'd gear down to about 15-to-1 to limit the current draw and protect the speed controller. I could probably build an electric powered SHW pushybot for around $3,000 to $3,500 with common off-the-shelf parts and steel. All of the tires and sprockets are supported on both sides with Zamak die cast pillow block bearings from Grainger, supporting 1" keyed shafts. The two batteries are just two 13 AH Hawkers. The tires are 10.5" diameter pneumatic tube type knobbys with split rims which came from Northern Tool and Supply. The six chains in the robot are made from #35 nonroller type.

The frame is built from 1" square tubing with a .065" wall thickness. The side rails of the frame are also supported by 1" schedule 40 steel water pipe that is braced with 1/2" schedule 40 water pipe. This produces a double triangle effect and gives the sides some protection. The front plow weapon is made from 1/4" mild steel plate. Evelyn has been the most successful robot I have ever built; it has won 16 out of 19 fights and four heavyweight titles."

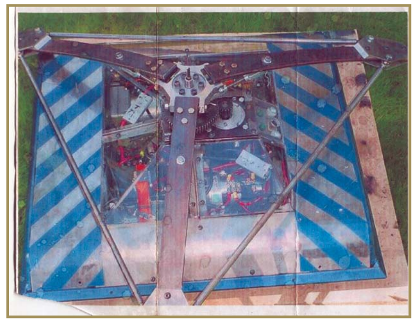
Basic Machines



Baby Boomer started life as an "eBay bot."

Builder John Culleton, from Team Boom Bots, builder of Heavyweight "Baby Boomer," calls himself an eBay builder:

"I really cringe at paying $800+ for motors, $800+ for speed controllers, $400+ for R/C control etc. ... I like just surfing around and getting odd stuff off of eBay and other places. I got one 120V DC motor for $40, I have six wheel chair motors that cost $80/pair, a couple of old scooter motors for $80 a pair, and then I finally bought Mag C40-300 and eventually Mag S28-400s at big dollars. Entry (first time bots) need to be cheap so the person doesn't feel so scared to get the bot destroyed. I built 3-4 HW robots that never got finished or didn't compete with the cheap parts until I started trying to do it right. My first bot was 2 x 6 steel tube with a Nissan crankshaft for a weapon. It shook like crazy but scared me enough to get me hooked."



Heavyweight LNW was built for around $1,000.

Greg Schwartz from Team LNW relates that his first big bot was very basic. His story:

"My son and I were hooked watching BattleBots, and never would have built a HW, until I stumbled across a wheel chair repair business. I talked to a guy named Ray, asked some questions, told him what we had in mind, and walked out with six used four-pole right angle chair motors free, plus some other goodies like tires, etc. We went

Team LNW's Heavyweight Parts	Cost
Steel, axles, bearings, gears, sprockets, weapon shaft (surplus w/employee discount)	$170
Invacare model 1085952 wheelchair motors (from "Ray")	Free
Two Victor 883 controllers, R/C switch, solenoid	$360
Four sealed lead acid batteries, 12V, seven amp hour	$140
C40-300 Magmotor (used)	$125
Titanium and Polycarbonate (surplus)	$55
Leaf Springs (used)	$20
Misc. nuts, bolts, washers, etc. (at cost, estimated)	$60
Machining	$70
Total:	$1,000

Table 4. Team LNW Costs.

through two seasons with these, Victors, SLAs, a used Futaba, a ton of mild steel tubing, and a ton of hope. It cost us around $700 for our first competition."

He supplied a materials list and cost estimate, which is shown in Table 4.

Asked about removing the weapon cost, Greg figures the $1,000 estimate could be reduced by $250 or so, leaving a ballpark figure of about $750 for a pushybot. As far as combat reliability, here's the story of the first version of LNW in the box:

"Our first competition was against Shrederator (very poor luck of the draw) at WBX. About the third hit, he somehow made our weapon rise up, then shear off our antennae ... also warped the spinner frame beyond repair. Our second competition was against Brick, with a makeshift wedge and no spinner. That one lasted about 0.5 seconds ... Didn't even have time to turn and get out of his way. So from lessons learned, our new LNW has a vertical spinner, no tracks, and no antennae up through horizontal spinners."

Conclusions

It must be remembered that fighting a robot costs more than just building the bot. As mentioned, a radio system must be procured, and also battery chargers, spare parts, shipping bots to events, shipping the debris home, tools, entry fees, and travel all enter into the total cost of the sport. Still, my research shows that established builders can get into the heavy weight classes for as little as $1,000, and not more than $5,000, a far cry short of the "as much as $20,000" figure that's scared this author away from building a SHW bot. Of course, now we have to repeat the "pushy" vs. "weapon" argument, select a radio system, talk about shipping disasters, machined vs. bolted vs. welded frames.

See a later Combat Zone for some of these stories. Meanwhile, I'll be downtown at my local wheelchair store, looking for a guy named "Ray." SV

HARDCORE ROBOTICS Reveals the Secrets Behind Superheavyweight Contender TOMBSTONE

● by Kevin Berry, from information supplied by Ray Billings

Hardcore Robotic's Superheavy Tombstone is a great example of a low-cost, top performing bot (Photo 1). Ray Billings, team leader, offered to share with *SERVO* readers their techniques for accommodating these opposing factors. Just to establish credentials, according to Botrank (**www.botrank.com**), Tombstone is currently ranked #5 out of 16 active SHW (Superheavyweight), and stacks up historically at 11th, right behind world famous fighters like Shovelhead, Minion, Diesector, and Toro. Tombstone came in second at its first event (RoboGames 2004), won the 2004 National Power Chair Open, plus took another second at the 2004 Nationals.

Ray has this to say about their modest price tag: "I think it will shock you how cost-effective it can be to build a SHW if you want to. These are current costs today — I paid less than this when it was built." (See Table 1. Note: The cost of the radio system, including the onboard receiver, is not included in the table.)

Photo 1. Tombstone's (right) inaugural match at Robogames against Blue Max resulted in a big first win, but not unscathed!

Technical specs on the hardware reveal some of the compromises inherent in bot building:

- **Frame** — All made from 4130 steel. All one inch tubing, with thickness varying depending on the stress loads at that point of the bot. The weapon support arms take the most load, and are .188" wall thickness. Exterior supports are .090 wall, and internal supports are .060 wall. The support plates holding the weapon pin in place are 1/2" 4130 plate.

- **Weapon** — The weapon bar is S7 tool steel. Dimensions are 34" long, 7-1/4" wide, and 1-1/2" thick. Mounted to the hub assembly, the entire rotating mass weighs in at 97.5 pounds. Power is from an Etek at 48V, spinning the weapon at around 2,200 rpm.

- **Drive Train** — The drive train is a

little sluggish for a SHW. The NPC T-64s are adequate, and it's not like we're going to win any pushing matches anyway. Weight considerations drove this underperformance.

• **Batteries** — Weapon batteries are four Hawker PC545 13 AH batteries. The SLA style batteries are very heavy, but they can source a lot of amps, which is what it takes to get that big heavy bar up to speed quickly. The drive train is powered by a pair of 3.6 AH Battlepacks.

• **Armor** — "Dust covers" would be more accurate. All the armor on Tombstone is .080 aluminum. After the big weapon and all the batteries to drive it, minimizing weight elsewhere was mandatory.

• **Electronics** — The drive ESCs are Thors, and the receiver is Futaba. Controlling the switching of the weapon and providing the BEC is an RSGBX unit. This, in turn, switches a large marine contactor, which turns on the weapon motor. All of these systems are on separate and discreet systems, requiring their own master power switch. Also, LEDs are available to show RX power (yellow), drive power (green), and weapon power (red). As dangerous as this robot is, we want clear signals as to what is happening onboard.

The Hardcore Robotics website (**www.hardcorerobotics.com**) has a detailed build report on Tombstone. Excerpts and photos were supplied to *SERVO* for this article.

Take a look at Photos 2 and 3. The steel tubing will be pressed into the center hole in the bar. The aluminum hub will be pressed on the top portion of the steel tubing. Bolts will come up through the bar (small holes) into the aluminum hub (these holes aren't drilled in the hub yet), out through the top of the aluminum hub and hold the belt pulley (not shown) into place. The brass bushings will be pressed into each end of the steel tubing, and will rotate around a fixed shaft. That solid shaft is 1.5 inches thick. This all adds up to a fairly heavy mount system, but it should be strong enough to withstand the punishment it will be dishing out.

In Photo 4, you can see the NPCs mounted and where the Etek is going to be. The area behind the motors will be for batteries, and we should be able to fit all the electronics in front of the NPCs. With almost 100 pounds in spinning weight and 70 pounds in batteries, it's going to

Component	Component Price	Total Price
NPC T-64 Gearmotors (two)	$286	$572
NPC tires (two)	$73	$146
Etek motor	$450	$450
Etek motor mount	$55	$55
Hawker PC-545 (four)	$90	$360
24V 3600 mAH NiCAD Battlepacks (two)	$168	$336
Victor 885 speed controllers (two)	$200	$400
Blue Sea Solenoid Switch E-series 9012	$130	$130
RSGPX Battery Eliminator/Switch Controller	$57	$57
Frame and Armor		$300
Weapon System (bar, pulleys, belts, bearings, hub)		$1,000
Total:		**$3,806**

*Prices (in part) from **www.robotcombat.com**, **www.battlekits.com**, **www.battlepack.com**, **www.ifirobotics.com**, and **www.roboticsportinggoods.com***

TABLE 1


Photo 2. A lot of machining time went into the S7 steel weapon bar.


Photo 3. Here is the bar with its mounting system.

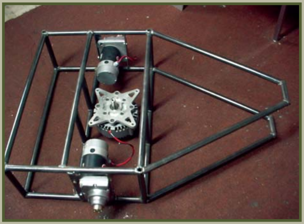


Photo 4. Start of the frame, showing the basic layout.

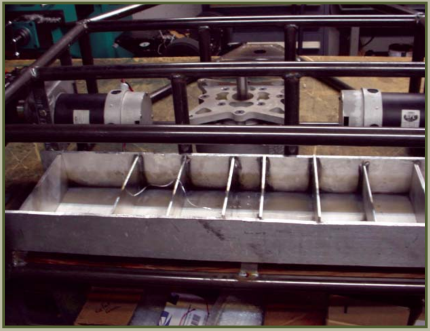
Photo 5. Battery box sitting in the frame.


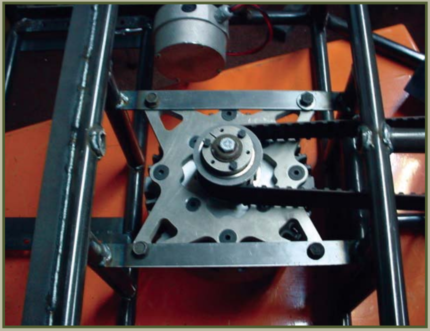
Photo 6. ETEK weapon motor in and mounted.


Photo 7. Test fit of the weapon system, also showing more progress on the frame.

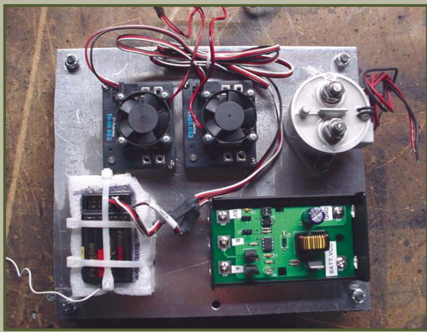



Photo 8. The electronics panel.

Photo 9. The frame is almost complete.

Photo 10. The finished bot, ready to enter the arena.

add up quick. The armor isn't going to be much!

If you look at Photo 5, you can see a small spot to put a couple of Battlepacks on the left. These will be used for the drive train, and only the weapon motor will use all the Hawkers (see Photo 6).

Photo 7 shows the test fit of the weapon system. The weapon looks like it will be perfect as far as clearances go, with the point on the blade about three inches from the floor.

We decided to mount all the electronics on one panel, and use some rubber shock mounts to keep vibration and impacts from damaging stuff (see Photo 8). We'll see how well it works out — not sure the Rx (lower left) is going to like being right next to those ESCs. The contactor in the upper right of Photo 8 is a marine unit, capable of 2,000 amps inrush. You can see the two Victors in the upper left and the unit in the lower right is a RSGBX.

At this point, the frame is almost complete. Look at Photo 9 — you can see all the internals are removed, to drill and tap the armor mounts. The armor panels are in place in almost all areas, and most of the gusseting and supports are in, as well.

And ... it's finished! You can see almost in the very center of Photo 10 we added an idler to keep tension on the belt. We'd underestimated how much torque this motor has, and even though that's a 1.5 inch wide belt, it would stretch far enough that it would jump the teeth on the pulley. Other changes that have come about along the way include the small bevel on the front of the frame. This will be the drag point on the front, keeping the bottom of the weapon shaft pin from getting drug and scuffed.

Overall, this bot ended up being overweight and we ended up making the decision to pull one of the Hawkers and run the weapon at 48 V. This brings Tombstone down to almost exactly 340 pounds. Spin up time is around three seconds. The bot drives fine. SV

EVENTS

RESULTS — May and June

Mechwar 9 — May 20-21, Minneapolis, MN. Presented by Mechwars Robot Combat. Results are as follows:

● *Megaweights*: 1st: "MidEvil," pusher, Foaming Rampage; 2nd: "Jay," flipper, The Destroyers; 3rd: "Mangler," drum, Team Rusty Nuts.

● *Superheavyweight*: 1st: "Psychotic Reaction," spinner, Kontrolled Kaos; 2nd: "Star Hawk 3.0," Spinner, Team Moon; 3rd: "Stump Grinder," flail, Team FUBAR.

● *Heavyweights*: 1st: "Shrederator," full body spinner, Team Logicom; 2nd: "Eugene," spinner, Team Moon; 3rd: "TY," Team Bobbing for French Fries.

● *Middleweights*: 1st: "Al," flamethrower, Team Bobbing for French Fries; 2nd: "Maxo," spinner, RoboRedNecks; "Bot Named Sue," flap & saw, Robocommand.

● *Lightweights*: 1st: "Bob," flamethrower, Team Bobbing for French Fries; 2nd: "Llamakazi," wedge, Fab Lab Robotics.

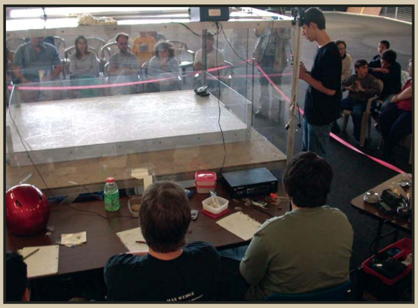
● *Featherweights*: 1st: "The F-Bomb," spinner, Killerbotics; 2nd: "DOA," spinner, 564 Robotics; 3rd: "Buggy Debug," pusher, Killerbotics.

● *Hobbyweights*: 1st: "KITT," wedge, Team Moon. Beetleweights "Fire Fly," wedge, Team Booyah; 3rd: "Thanatos," full body spinner, Team Python.

● *Antweights*: 1st: "ANTI," spinner, 564 Robotics; 2nd: "Doorstop," wedge, Team Falcon; 3rd: "UnderWHERE?," spinner, Hazardous

Robotics.

House Of Robotic Destruction — May 20th, Olmsted Falls, OH. Presented by the Ohio Robotics Club



HORD Spring brought in spectators and contestants from across Ohio and nearby states to enjoy the spectacle of combat robotics! This RFL Nationals qualifying event used a great insect arena that has two pits that open during a match. Results are as follows:

- *Antweights*: 1st: "Flipper," a flipper, Team Hoff; 2nd: "The Froogin," wedge, Team FishNecks.
- *Beetleweights*: 1st: "One Fierce Uppercut," vertical spinner, Fierce Robotics; 2nd: "One Fierce Lawnboy," egg beater, Fierce Robotics.
- *Ant Rumble winner*: "Karl Marx Hoards Candy," wedge/scoop, All Things Must Die.
- *Beetle Rumble winner*: "One Fierce Lawnboy," egg beater, Fierce Robotics

Gilroy Bot Gauntlet — May 27th, Gilroy, CA. Presented by California Insect Bots. Two Fleas, nine Ants, and four Beetles competed. Results are as follows:

- *Fleaweights*: 1st: "Ugly Duckling," Lifter, Team Slayer; 2nd "Change of Heart," wedge, Team Misfit.
- *Antweights*: 1st: "Front Kick," wedge, Team Kick-me; 2nd: "Ducbot," lifter, Team Slayer; 3rd: "Fire Eagle," wedge, Team Misfit.
- *Beetleweights*: 1st: "Unknown Avenger," flipper, Team Ice; 2nd: "Bite Me," 3rd: "Toe Poke," Lifter, Team Kick-me.

Spring Whyachi House of Robotic Entertainment 2006 — May 27th, Dorchester, WI. Presented by WHRE. Eleven bots participated. Results are as follows:

- *150g*: 1st: "Kankle Killer," saw blade, Team Whyachi; 2nd: "Paper Cut," vertical spinner, Iron Fist Combat Robotics; 3rd: "Micro Brick."
- *Antweights*: 1st: "Nano Falcon," drum, Team Whyachi; 2nd: "KILLER Aluminum Sandwich," horizontal spinner, Iron Fist Robotics; 3rd: "ANTI," vertical spinner, 564 Robotics.
- *Beetleweights*: 1st: "Little Brick;" 2nd: "Mini Munch;" 3rd: "Burger Time." SV

TECHNICAL KNOWLEDGE

Scrounging Parts

● by Kevin Berry

There are quite a few online stores that sell robot parts. However, the naturally creative bent displayed by builders, coupled by limited budgets, leads to some pretty inventive uses of handy materials in combat bots. My first lightweight's frame was made from the legs on discarded industrial shelving. My first two "insect" bots resided under baking pans. Even though I've graduated to high-end, specialized items, I still find myself pouncing on odd brackets, shafts, and gears left over from home improvement projects or disabled appliances.

The absolute number one scrounge in combat robotics is the cheap, "Harbor Freight" drill motor. These have powered innumerable bots since the sport was invented. A fresh shipment of drills prompts someone to buy one and sit in their car in the parking lot, checking for metal gears instead of plastic. A "Eureka" leads to buying a case of drills, then hawking them on combat forums to everyone's delight. I put an ad in Kennedy Space Center's weekly bulletin, asking for dead, battery-operated power tools, which yielded more than I can ever use, mostly for free.

Several insect bot builders get most of their material from old copiers. Gears, chains, belts, sprockets, shafts, wheels, and fasteners abound. VCRs, DVD players, and paper shredders are great sources for gears and springs. Portable and cell phone batteries, while very low current, can supply some micro bot needs. Another very popular source for starter insect bots are hacked R/C toys, particularly the "BattleBot" series. There's a whole box of dead toys in my shed, waiting to be used by newbie builders. One Florida builder swears by Megablocks BattleBloks, and has produced a national qualifying beetleweight with them.

For bigger bots, some entry level builders use electric riding toys, such as "Powerwheels" for drive components. I've been seen pulled over by the side of the road on garbage day, swiping the motor/gearbox combos out of some child's dead and broken Barbie Jeep.

More medium-sized bot materials can be obtained from computer cases, sides of filing cabinets, and steel shelves. Old bed frames are a great and free source of angle iron (but a bit hard to cut and weld). Rebar can be used for rod, although a coat of paint is needed to prevent rust. Leaf springs are commonly used for bumpers or spinning weapons.

Lawnmower and wheelbarrow wheels are sometimes used and, let's face it, who can throw away a perfectly good wheel? Every backyard mechanic has a pile of them somewhere.

And, let's not forget the number two most popular scrounge in the sport — the famous EV Warrior style motor, direct descendent of power window and windshield wiper motors pulled from junk cars and sent to do battle in the box.

One builder described the classic junk combat bot:

"On my team's first middleweight, we used almost no purpose-built robot parts. Our armor was tin made for roofs, our weapon was a broken sledge hammer, our motors were from a treadmill, our chains and sprockets were from a bike, and my favorite: our wheels were from a ball return system from a bowling alley. Besides the fact that they were 8" tall and about 4" wide and had no good way to mount to anything, they had a ton of grip."

Not all the items mentioned are perfectly suitable in a combat environment, but there are national level bots employing all these ideas. Combat robotics is a "learn by doing" sport, and starting off cheap and creative leads to a real understanding of where a builder can economize, and where to put money and time in custom, or expensive parts. Thanks to the builder community for these ideas! SV

PRODUCT REVIEW — BaneBots Gearmotors

● by Michael Vroegop, Berserk Robotics

BaneBots is one of the newest combat robot parts suppliers around, and one of their main products is a line of inexpensive gearmotors. They're sized according to their gearhead diameter, ratio, and the type of motor powering them. The ratios are:

● 16 mm — 11:1, 24:1, 38:1, 54:1, 118:1, 574:1

● 25 mm — 10:1, 20:1, 30:1, 40:1

● 28 mm — 5:1, 16:1, 20:1, 25:1, 64:1, 100:1, 256:1

● 36 mm — 5:1, 16:1, 20:1, 25:1, 64:1, 100:1, 256:1

In addition, they come with these motor sizes attached:

● 16 mm — All FF-050. Very small, like long-canned servo motors.

● 25 mm — FF-180 and RF-370.

● 28 mm — All RS-385 motors, comparable to speed 400.

● 36 mm — RS-385 and RS-545 motors, similar to large cordless drill motors.

It's quite a wide range of gearmotor sizes — essentially, you could equip a full sub-lightweight fleet with nothing but orange-stickered BaneBots drive motors. I've used the 11:1 16 mms and the 20:1 25 mms with ff-180s and they performed well — the 16 mms worked well enough to take home 1st place at the recent Battle Beach competition in Florida in my antweight Peligro.

The most common failure mode for Banebots seems to be burning out motors — with the FF-050, I've gotten away with 11.1 V for the most part (8 V max rated). On the other hand, the FF-180s in the 25 mms die at 11.1 V quite rapidly. This is user error, however, since their maximum rated voltage is 4.5 V!

Construction of the 16 and 25 mm gearmotors is sturdy enough, their gearhead cases are coated steel, and gears are brass, strong enough with impact absorbing, foam rubber wheels. One weak spot which is well-documented is the overhung output shaft — it's recommended that the ends of the shafts be supported by an outer frame plate or similar, which I found unnecessary at their small scale. Shafts are common sizes, I was able to use my 3 mm prop adaptors on the 16 mm ant-sized motors and my usual 4-40-pin method on the 4 mm shafts of the 25 mm beetle-sized motors.

The greatest thing about the BaneBots is their price. Simply put, they perform as well as Copals or B62s, and cost roughly half of the rarer alternatives. They're all I plan to use on ants and beetles in the future.

Visit the BaneBots website at **http://banebots.com** SV

EVENTS

UPCOMING — August and September

Robot Fighting League Nationals — August 11-13, Minneapolis, MN. Presented by the Midwest Robotics League.

The 2006 Robot Fighting League
Nationals
Presented by
Midwest Robotics League
GibbsCAM Powerfully Simple. Simply Powerful.
Exact JobBOSS THUNDERDOME

This event culminates the robot fighting season. Includes a "last chance" qualifier on the first day, open to previously unqualified bots. This is THE event of the year. All the best teams and bots come together in a melee of destruction. If you go to one event this year, this is the one. Visit **www.kickbot.org** or **www.botleague.com** for more details.

The Texas Cup — September 9th, Carrolton, TX. Presented by Southwestern Alliance of Robotic Combat.

SWARC
WWW.SWARC.ORG

Classes from 150 grams up to 120 pounds. Venue is Mike's Hobby Shop (**www.mikeshobbyshop.com**). Spectator admission: $2, limited seating. VIP passes required for restricted area overlooking arena. Registration limited to 16 bots in each class. Prizes: First and Second place only. Medallions will be awarded. Sponsorship certificates will be awarded. Format: Standard double elimination, all classes. This is a 2006 qualifier for the RFL Nationals. Visit **www.robotrebellion.net**

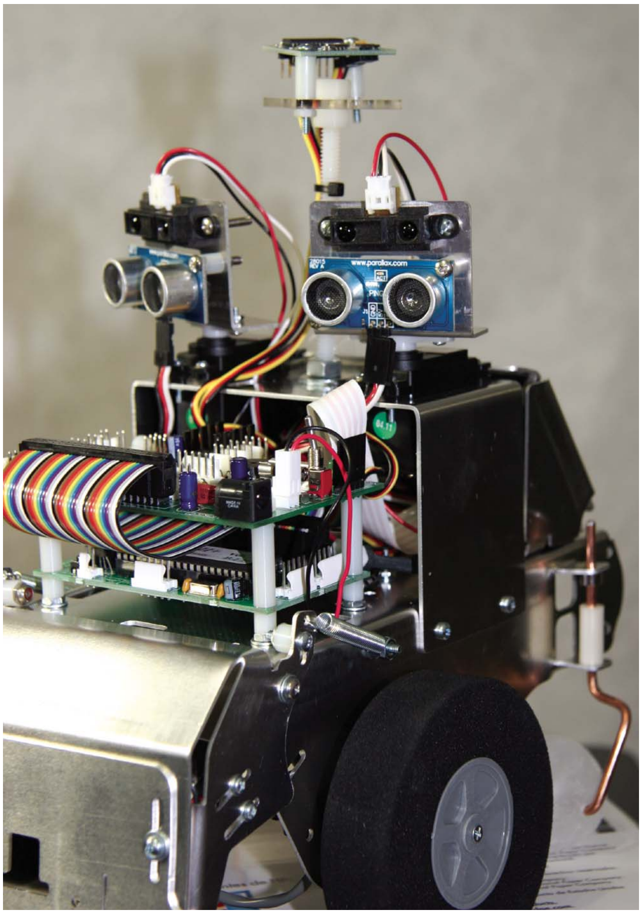
Fall Whyachi House of Robotic Entertainment 2006 — September 16-17, Dorchester, WI. Presented by WHRE.

No pit passes, no limits on pit members, no fee for spectators, all entry fees put into prizes and cash for competitors. **SV**

W.H.R.E.
Whyachi House of Robotic Entertainment

AUTONOMOUS ROBOTS

and Multiple Sensors



Part 2:
ADVANCED SENSOR FUSION

Part 1 revealed that sensors are imperfect, provide limited data, and that their performance is a function of the operating constraints imposed by the environment. Furthermore, although data from individual sensors can be handled independently, doing so forfeits potential gains in accuracy, completeness, performance, and dependability. This article continues the discussion of sensor fusion — the simultaneous use of data from multiple sensors — in autonomous robotics applications.

CONTEXT

Because sensor characteristics vary over time, fusion of data from a single sensor can enhance data quality. This is the rationale for the common practice of averaging the data from the analog IR rangefinder GP2D12. However, single sensor fusion based on simple averaging of sensor data doesn't reduce the uncertainty of measurement or systematic errors due to, for example, a damaged sensor. Also, a robot that relies on a single sensor has a less complete measure of the environment, compared with a robot equipped with multiple sensors.

Whether multiple sensor fusion is implemented at the signal, data, feature, or decision-making level, it occurs in the context of a specific logical and physical sensor arrangement and mix. A cluster of sonar rangefinders arranged to provide 360 degree coverage in the horizontal plane is an example of *complementary* fusion. This method of ensuring completeness of data is usually preferable to the less expensive approach of using a single sonar rangefinder mounted on a servo, especially in a rapidly changing environment.

Sensor fusion is often used with data from multiple sensors of different types that measure the same parameter. This *competitive* sensor fusion, exemplified by the common use of both ultrasonic and IR rangefinders for obstacle avoidance and environmental mapping, can address uncertainty, systematic errors, and sensor failure. Figure 1 illustrates a sensor architecture that relies on both cooperative and complementary fusion to support autonomous behavior.

Assuming optimal placement, failure of two of the three sensors should result in graceful degradation of robot navigation or tracking performance, as opposed to total failure. Note that cooperative fusion of the sonar rangefinders occurs at the signal level, while competitive fusion of IR and ultrasound rangefinders occur at the data level.

Approaches to sensor fusion

PHOTO ABOVE. Electronic compass, ultrasonic and infrared rangefinder sensors on a carpet rover.

NOTE

The code listings mentioned in this article are available on the *SERVO* website at **www.servomagazine.com**

include a variety of elementary statistical methods, probabilistic or Bayesian methods, fuzzy logic, and more advanced methods, such as the Kalman Filter. Hundreds of variations of these and other approaches — used singly and in combination — have been developed because no methodology is clearly superior in every situation.

Sensor fusion that relies on elementary statistical operations such as mean, mode, and median, is relatively easy to implement at the signal and data levels. Fuzzy logic methods assign sensor data values to membership in predefined, fuzzy groups. For example, a target that isn't detected by an ultrasonic rangefinder until it is close to the sensor is "probably" a soft rubber ball. Fuzzy logic and Bayesian methods are often used at the feature and decision-making levels. Statistical methods and the popular Kalman Filter are discussed in more detail here.

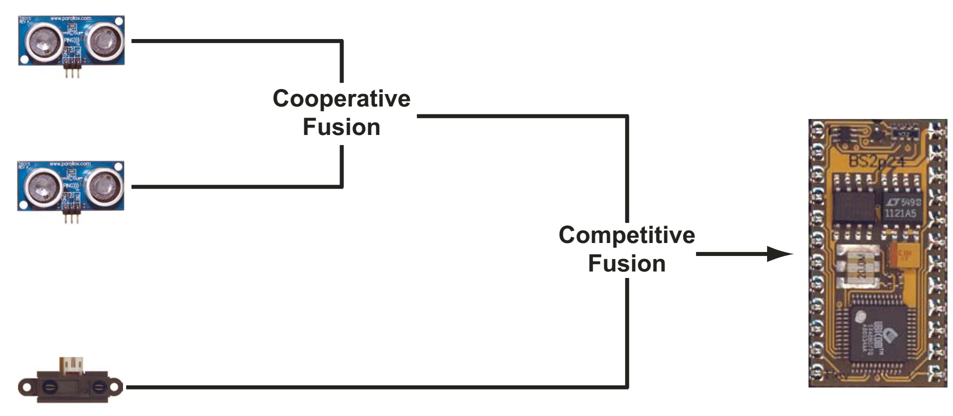


FIGURE 1. A mixed-sensor architecture with complementary and competitive sensor fusion.

ELEMENTARY STATISTICAL METHODS

The statistical mean of a series of sensor measures is frequently used in cooperative fusion to reduce data variability. Although simple to implement, the mean is very sensitive to outliers that are not distributed symmetrically around the mean. For example, a single, abnormally high sensor reading can shift the mean to a higher value. As a result, the range of uncertainty of sensor data isn't reduced. A more powerful approach to statistical sensor fusion is to use a statistical method that considers specific sensor characteristics, such as the *weighted mean*.

The weighted mean is the sum of the weighted scores over the sum of the weights, calculated as:

$$Weighted\ Mean = \frac{\sum (w^i x^i)}{\sum w^i}$$

where w^i is the weight corresponding to a data value x^i. The benefit of using the weighted mean over the simple mean in sensor fusion is that more accurate sensor data contribute more to the measure than less accurate data. That is, weight is given to sensor data in proportion to its accuracy.

Consider a sensor configuration like that of Figure 1 in which the distance data from the IR rangefinder is more accurate than the fused data from the ultrasonic rangefinders, by a factor of 2-to-1. Assuming the distance as measured by the IR rangefinder is 30 cm and the distance measured by the ultrasonic rangefinder is 40 cm, the simple mean is 35 cm. In contrast, using the weighted mean, in which weights are assigned by relative merit, the distance is:

$$Weighted\ Mean\ Distance = \frac{(30cm \times 2) + (40cm \times 1)}{2+1} = 33.3cm$$

The weighted mean has little computational overhead and is useful for competitive fusion at the signal and data levels, especially when sensors produce data at different rates. A limitation of a sensor fusion algorithm based on the weighted mean is that the technique doesn't automatically degrade gracefully with sensor damage or failure. Returning to the scenario depicted in Figure 1, if the IR rangefinder returns a value of 30 cm and the ultrasonic rangefinders return an out-of-range value, e.g., 369 cm for the Parallax Ping)))™ rangefinder, then the weighted mean of distance becomes:

$$Weighted\ Mean\ Distance = \frac{(30cm \times 2) + (369cm \times 1)}{2+1} = 143cm$$

The same overestimation of distance occurs when one sensor detects an object before the other sensor. Provision for ignoring the out-of-range sensor data can be made by dynamically adjusting the sensor weight used in the numerator and denominator to zero, as in:

$$Weighted\ Mean\ Distance = \frac{(30cm \times 2) + (369cm \times 0)}{2+0} = 30cm$$

The code snippet in Listing 1 illustrates a subroutine in PBASIC for the BASIC Stamp that can fuse data from two rangefinder sensors using the weighted mean. When instantiated with the maximum in-range values and relative weights for each sensor, the subroutine *FuseData* returns the weighted mean of the two sensor values. If data from a sensor is out of range, then the weight associated with data from that sensor is set to zero. The routine can be extended to work with any number of sensors within the I/O and memory limits of the microcontroller.

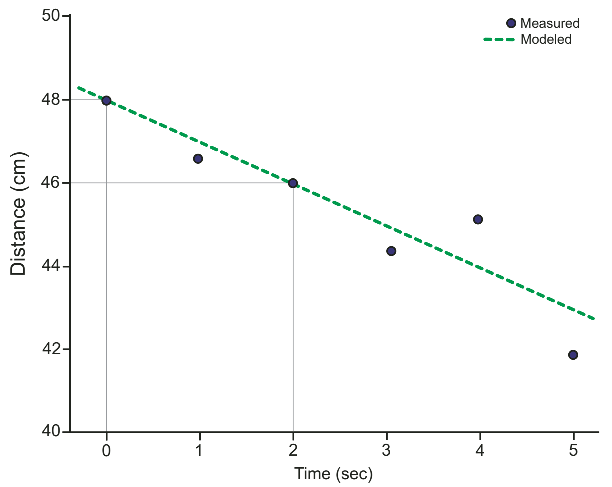
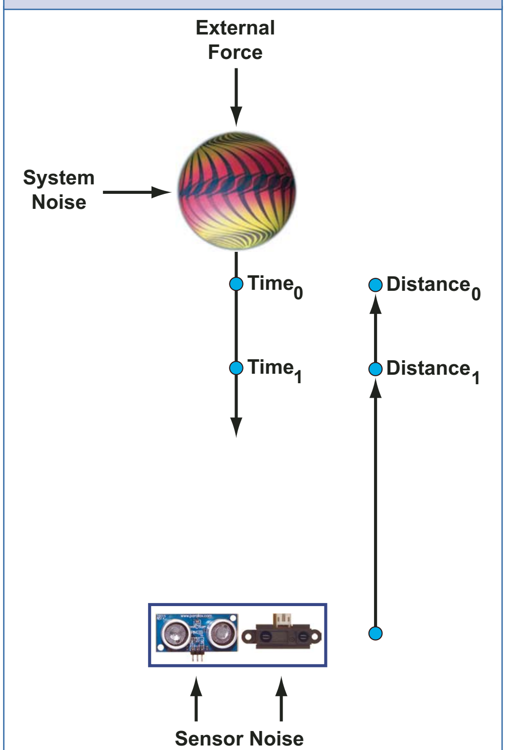


FIGURE 2. Time versus distance based on measured and virtual sensor data.

The routine is agnostic to sensor sequencing, in that the sequence of reading data from the sensors isn't specified. It's assumed that the variables *Sensor1* and *Sensor2* are assigned sensor data in a timely manner. However, for rapidly changing sensor data, sequencing can be critical. A significant time interval between each sensor update can be a significant source of error.

FIGURE 3. The ball subject to an external force. Note the noise on the sensors and the ball.



THE KALMAN FILTER

Sensor fusion based on elementary descriptive statistics, such as the mean and weighted mean, can provide a robot with a historical perspective of its environment. In a static environment, historical data may be good enough. However, an autonomous robot in a dynamic environment can often benefit from an ability to accurately predict the future — a characteristic of intelligent organisms [1].

An advanced statistical method that can be used to predict future sensor data, and therefore the future environment, is predictive modelling. The accuracy of prediction depends on how far out into the future the prediction is made, the time-varying nature of the sensor data, and the accuracy and quality of the sensor data used in the prediction. The farther out in time, the more erratic, and the poorer the data quality, the lower the prediction accuracy. A review of predictive modeling is relevant here because it provides the foundation for a discussion of a popular, advanced sensor fusion methodology — the Kalman Filter.

Predictive Modelling

Predictive modelling involves defining a function or model, generating virtual sensor data with the function, and observing how closely the virtual sensor data match the real sensor data. If there is a significant difference, the parameters of the model are modified accordingly. The process is repeated until the virtual and real data are close enough, as defined by some objective criteria.

Consider the linear trajectory of a rubber ball rolling on the floor after it has been kicked toward our robot by an opposing robot soccer player (see Figure 2). Based on sensor data at 0 and 2 seconds, the ball is approaching our robot at a steady-state, linear velocity of 1 cm/second. The model of the ball, in terms of distance from our robot versus time, can be expressed as:

$$s = k - v_w t$$

where s is distance from the sensors on our robot in cm, k is the initial sensor-to-ball distance, v_W is the weighted velocity of the ball in cm/sec, and t is time in seconds. The weighted velocity is calculated with the weighted mean distance reported by the two sensors, as described earlier. Assuming the weighted velocity is 1 cm/sec, the model describing the distance of the ball from our robot becomes:

$$s = 48 - 1cm/sec \ x \ t$$

A potential advantage of using this formula instead of relying solely on subsequent sensor data is that, assuming the model is correct, meaningful data on the state of the ball is available despite missing and inaccurate sensor data. Furthermore, assuming steady state, data can be derived from the model for any time in the future. For example, at five seconds, the ball should be 43 cm from our robot. Unless our robot or the ball change trajectories, impact will occur 43 seconds later.

As a check on the validity of the model, distance could be checked every four seconds. If there is a deviation from the model, the formula could be adjusted accordingly. In the real world, the ball would be subject to outside forces or factors in the environment, such as rolling resistance, that would change the velocity over time.

Now consider the situation in which initial sensor distance measurements are taken during the time the soccer robot's foot is in contact with the ball. That is, the ball is subject to acceleration (see Figure 3). The rubber ball is heading straight to our robot, but the relationship between distance and time is *nonlinear*.

Given this new situation, the previ-

ously defined linear model is insufficient to accurately predict the future location of the ball. Although a linear approximation may be good enough for small intervals — say, to describe the distance travelled between 2.0 and 2.5 seconds — a second or third degree polynomial is needed to accurately model the ball. Recall that an n^{th} degree polynomial takes the form:

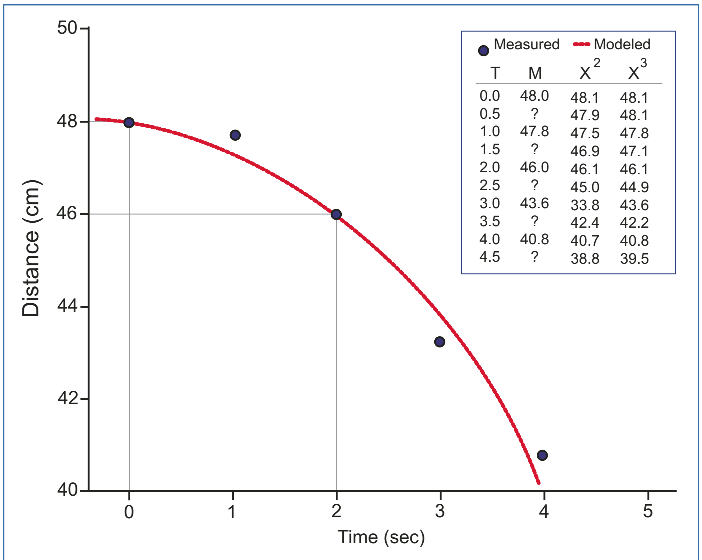
$$p(x) = p_1x^n + p_2x^{n-1} + p_3x^{n-2} + ... + p_nx + p_n$$

Figure 4 shows the sensor data and the resulting curve fitted to the data using the polynomial curve fitting function within Matlab. The formula for the third degree polynomial, which is used to generate the curve in Figure 4, is:

$$p(x) = 0.1x^3 - 1.0143x^2 + 0.6571x + 48.0114$$

Matlab was also used to generate a second degree polynomial:

$$p(x) = -0.4143x^2 - 0.2029x + 48.1314$$



T	M	X^2	X^3
0.0	48.0	48.1	48.1
0.5	?	47.9	48.1
1.0	47.8	47.5	47.8
1.5	?	46.9	47.1
2.0	46.0	46.1	46.1
2.5	?	45.0	44.9
3.0	43.6	33.8	43.6
3.5	?	42.4	42.2
4.0	40.8	40.7	40.8
4.5	?	38.8	39.5

FIGURE 4. Non-linear curve fitting using third degree polynomial (X^3). Second-degree polynomial data are also listed (X^2).

Although less computationally demanding, the curve produced by the second degree polynomial doesn't fit the measured data, as well as the curve defined by the third-degree polynomial.

Curve fitting — the process of defining and iteratively refining a nonlinear equation to fit the available data — may be based on trial-and-error, or the use of analytical tools, such as Excel, SPSS, or Matlab. These and similar tools commonly rely on a "least squares" strategy to define the function that best describes sensor data.

Least squares attempts to minimize the sum of the squares of the ordinate (y value) differences between measured data and data produced by a formula that describes the data. Mathematically, using the least squares method, the goal is to define a polynomial such that the sum of the squares of the differences between actual and computed values are minimized:

$$Sum\,Of\,Squares = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

where y_i is the measured value and $f(x_i)$ is the corresponding calculated value.

In predictive modelling work, a commonly used variant of the basic least squares method is *weighted* least squares. As with the weighted mean, more weight is assigned to more trusted data. Mathematically, weighted sum of squares is computed as:

$$Sum\,Of\,Squares_{Weighted} = \sum_{i=1}^{n} w_i (y_i - f(x_i))^2$$

where w_i is the weight assigned to the corresponding data y_i. The overhead of a weighted least squares algorithm for nonlinear curve fitting is beyond the capabilities of typical microcontrollers used in hobby robots. Regardless of the hardware platform, an underlying assumption of both weighted and unweighted least squares is that errors in sensor data are randomly distributed. This turns out to be an important assumption in many advanced sensor fusion algorithms.

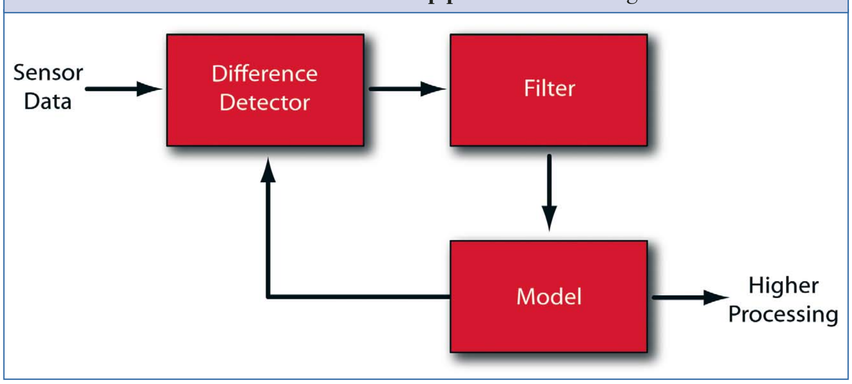
Given a weighted least squares predictive model of our ball, we can create a closed loop system that iteratively compares the model results with actual sensor data and corrects the model accordingly (see Figure 5). Model output, not sensor data, is used by the robot as the basis for higher level processing. Furthermore, the feedback isn't in the form of the sensor data, such as distance in cm for a rangefinder sensor, but appears as one or more of the parameters used in a polynomial model of the ball.

Least squares is a powerful technique. Variants of the least squares algorithm form the basis of many digital signal processing (DSP) routines, such as active filters. However, using ordinary least squares to iteratively modify a higher-order polynomial in real time is computationally intensive, and there is no guarantee that the results will be optimal.

The Classic Kalman Filter

The Classic Kalman Filter (KF), which can be thought of as a weighted least squares predictive modeling algorithm on

FIGURE 5. Closed-loop predictive modeling.



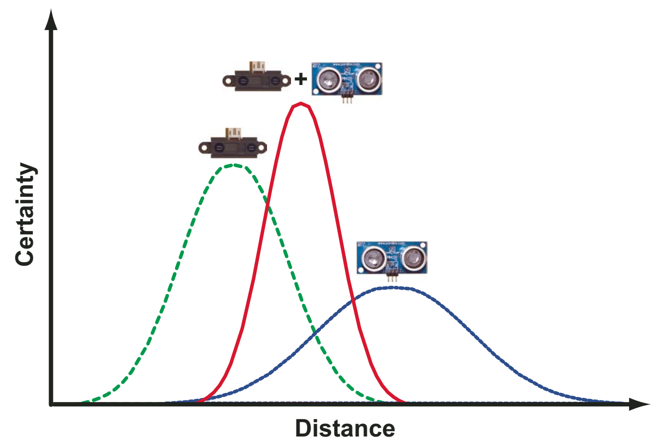


FIGURE 6. Probability density of sensor data fused by a Kalman Filter is greater than that of individual sensor data.

steroids, is the most popular of the advanced methods of sensor fusion. The algorithm considers the noise and uncertainty in sensor data and the statistical characteristics of the underlying model in recursively predict future sensor data [2]. As in the approach to predictive modeling described above, the algorithm makes a prediction of the future state and recursively corrects the prediction with imperfect sensor data. The amount of correction is a function of the difference between the actual and predicted sensor data and the quality of the sensor data.

When properly implemented, a Kalman Filter can fuse sensor data in a way that reduces uncertainty, as depicted in Figure 6. This is the magic of the Kalman Filter. Fusing data from multiple sensors with a Kalman Filter can produce data with greater certainty (lower variance) than data from the contributing sensors. Furthermore, this information gain is possible even when the data contributed by individual sensors is of poor quality.

The Kalman Filter is a closed-loop, recursive algorithm that operates on samples of sensor data at discrete time intervals. An important distinction is that — unlike many other sensor fusion methods — the algorithm doesn't require a database of all prior data values. Current sensor data are used to predict sensor data at the next time interval. Historical sensor data are discarded.

In the original or Classic Kalman Filter algorithm, three linear equations must be solved: the State Estimate, Kalman Gain, and Estimation Error Covariance. Ignoring the obtuse nomenclature and the matrix manipulations for this discussion, the State Estimate equation is of the general form:

$$State\ Estimate_{T+1} = [State\ Estimate_T] + K\ [Sensor\ Update]$$

The *State Estimate*$_{T+1}$, the model at time *T+1* in the future, is based on the status of the model at the current time, *State Estimate*$_T$, plus an adjustment based on the Kalman Gain, K, and a correction factor based on new sensor data that we'll call the *Sensor Update*. In an ideal world, with a perfect model and perfect sensors, the model state is updated with errorless sensor data with each iteration of the algorithm.

The second linear equation of the filter, the Estimation Error Covariance, reflects how well the actual and modeled data co-vary or track together. The third equation, that of Kalman Gain, *K*, determines the effect of newly acquired sensor data on the State Estimate. Kalman Gain is proportional to the quality of signal data, and inversely proportional to sensor noise. If the sensor noise is high, then K will be low, and the new sensor data will have little to no effect on *State Estimate*$_{T+1}$. Conversely, if sensor noise is low, *K* will be high, and the update will have a significant impact on *State Estimate*$_{T+1}$.

In this simplified view of the Kalman Filter, given the model of our rubber ball, updating the model parameters by incrementing the time to some point in the future should give the future state of the ball. For example, if the ball is traveling toward our robot at 1 cm/sec, at one second in the future, the ball should still be traveling toward our robot at 1 cm/sec. We would expect the rangefinder reading to decrease by 1 cm.

A state estimate or model that includes a prediction of the velocity of the ball takes the form:

$$Velocity_{T+1} = Velocity_T + Acceleration_T \ x\ Time + Noise_{Velocity}$$

The distance between the robot sensor and the ball at one time step in the future is equal to the current distance, less the distance covered at the current velocity during one time increment, less the contribution of acceleration. *Noise*$_{Velocity}$ may include minor wind drafts, the perturbations caused by myriad factors ranging from imperfections in the floor, static electric attraction between the floor and ball, fluctuations in ball roundness due to barometric pressure, and even variations in the ball wall thickness.

The actual distance between the ball and our robot — *Distance Actual*$_T$ — is not directly measurable. Instead, sensor data is partially obscured by sensor noise, as in the following equation:

$$Distance\ Measured_T = Distance\ Actual_T + Noise_{Sensor}$$

Noise$_{Sensor}$ could be attributed to factors such as temperature fluctuations, limited resolution of the A-to-D converter used with an analog sensor, or the physical characteristics of the sensor.

The Kalman Filter estimates and adjusts the values of variables and constants by comparing predicted and measured sensor data. The beauty of the Kalman Filter is that the algorithm automatically optimizes the underlying model by considering data from multiple sensors, the characteristics of the target and sensors, and virtually any other factors incorporated in the model.

To illustrate how the Kalman Filter can be used to fuse sensor data, assume that our robot is monitoring the distance of the rubber ball with both IR and ultrasonic rangefinders. In this scenario, the two sensors are polled in sequence, one second apart. Every second, data from the

other sensor is used to update the state of the model.

If ultrasonic sensor data are noisy, less accurate, or missing because of sensor failure, the Kalman Gain will be low, and the contribution of the ultrasonic sensor data to our State Estimate of the ball's velocity will be minimal. Conversely, if the IR sensor data suddenly became erratic because of dust on the sensor, then the contribution of IR sensor data to the model update will be minimal.

Assumptions

The cost of this power is complexity and the need to abide by three simplifying assumptions that make the Classic Kalman Filter computationally tenable:

1. The model or State Estimate can be defined as a linear function.

2. The model or system noise is white and Gaussian.

3. The sensor or measurement noise is white and Gaussian.

The first assumption — that the system being modeled is linear — is rarely true in practice. However, in robotics work, nonlinear systems can often be linearized or simplified with adequate results.

White, Gaussian noise is randomly and equally distributed over all frequencies. The Gaussian assumption means that the probability density amplitude of the noise is bell-shaped (see Figure 7) with a mean of zero. Note that noise may be Gaussian with a non-zero mean.

An example of non-white noise is a burst of noise that appears every three seconds, centered at the power line frequency of 60 Hz. Non-Gaussian noise includes noise with an amplitude distribution that is skewed, bimodal, or otherwise not bell-shaped.

Variations

Because the three assumptions are never completely realized in practice, hundreds of variations on the Classic Kalman Filter design have been developed since Rudolph Kalman published his paper describing the algorithm in 1960. Most of these variations provide workarounds for the assumption of linearity. Real-world, dynamic systems are more accurately described by nonlinear models.

The most common variations on the Classic Kalman Filter design include the Complementary Kalman Filter, the Extended Kalman Filter, and the Unscented Kalman Filter. The Complementary Kalman Filter is designed to estimate model errors, rather than the state of the model itself. In general, the Complementary Filter is more robust than the classic approach.

The Extended Kalman Filter is specifically designed to deal with non-linear models [3]. In exchange for increased complexity and greater computational overhead, the algorithm can accurately predict model behavior in spite of violating the first assumption of the Classic Kalman Filter. A simplifying assumption is that the mean value of the prediction is a function of the mean value of the input probability density function.

The Unscented Kalman Filter largely ignores all three underlying assumptions of the Classic Kalman Filter. Like the Extended Kalman Filter, the Unscented Filter can predict the state of nonlinear models. Moreover, because it can handle non-Gaussian noise, it is more robust than the Extended Kalman Filter. The price for this power is complexity and high computational overhead.

Performance

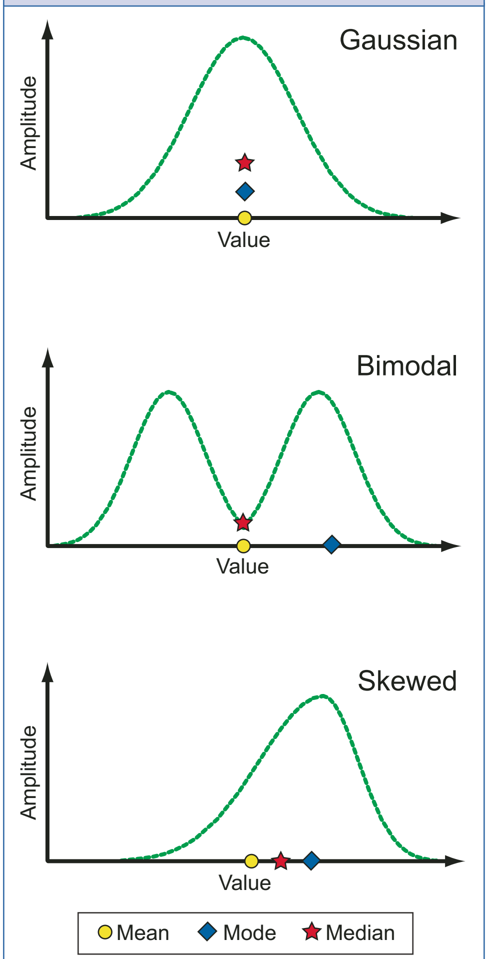
Used appropriately, the various forms of the Kalman Filter can fuse poor quality sensor data in a way that results in significant information gain. The Kalman Filter isn't perfect, however, given the underlying assumptions may not be valid in a particular scenario. Furthermore, the completeness of the underlying model limits the band of model certainty. For example, if robot sensors are tracking a rolling ball and suddenly a cat paws the ball, sending it off course, it's not likely that the Kalman Filter will be able to correct for the sudden change in the ball's trajectory. The Complementary Kalman Filter may be able to handle the change if it isn't too extreme.

An important issue in using the Kalman Filter to fuse sensor data is the computational overhead imposed by the need to manipulate matrices in real time. The more complex the model and operating environment, the larger the matrices. Because the computational overhead associated with model matrix operations is proportional to the cube of the matrix size, doubling the size of the matrix increases computational overhead by a factor of 2^3 or 8.

RESOURCES

Mathworld (**www.mathworld.wolfram.com**) offers an open reference on least squares and many other mathematical concepts related to sensor fusion. In addition, a great online resource on the Kalman Filter is maintained by the University of North Carolina at Chapel Hill, at **www.cs.unc.edu/~welch/kalman** The site includes an online learning tool written in Java, Matlab libraries, and C++ source code.

FIGURE 7. Examples of noise distributions: Gaussian, bimodal, and skewed.



REFERENCES

[1] Hawkins, J. and S. Blakeslee, *On Intelligence*. 2004: Times Books.

[2] Kalman, RE, A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME — Journal of Basic Engineering* (Series D 1960): pp. 35-45.

[3] Julier, S., J. Uhlmann, and H. Durrant-Whyte, *A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators*. IEEE Transactions on Automatic Control, 2000. 45: p. 477-81.

[4] Ding, Z., *Kalman Filter Implementation in the XA Eases Sensory Fusion*. Report AN715. 1999, Philips.

[5] Ristic, B., S. Arulampalam, and N. Gordon, *x*. 204: Artech House.

[6] Matlab. Commercial scientific computation and visualization. Requires the Control System Toolbox for the Kalman function. **www.mathworks.com**.

[7] Scilab scientific software package. A free, open source scientific software package. **Scilabsoft.inria.fr**

Fortunately, if the system is in steady state, only the State Estimate matrix has to be solved in real time. Even so, solving the Kalman Filter equations requires more computational power than provided by a typical microprocessor. As a point of reference, a Classic Kalman Filter implemented in C requires about 24K of code and 8K of data space [4]. This is within the capabilities of the New Micros ServoPod, Philips XA series, and ostensibly the Parallax Propeller.

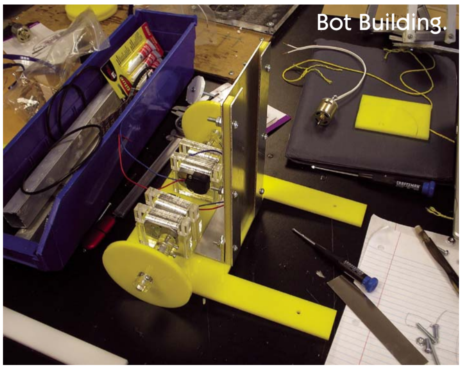
The Kalman Filter isn't the final word in sensor fusion. There is recent interest in the Particle Filter, which is based on defining the model in terms of a set of hypotheses called particles [5]. Whether this algorithm for nonlinear, non-Gaussian models will eventually replace the popular Kalman Filter remains to be seen. Regardless of the fusion method used, it doesn't obviate the need to carefully calibrate sensors, as discussed in Part 1. In fact, the gains from sensor calibration are often greater than those available through any type of sensor fusion technology.

HANDS ON

Obtaining an intuitive grasp of Kalman Filtering requires hands-on experience. One option is to explore variations of the filter using the commercial Matlab program and the Optimization Library [6]. A much less expensive option is to download the free Matlab-like scientific software package, Scilab [7]. Both packages feature a scripting language and graphing options that hide the complexity of matrix operations that form the basis of the filter. **SV**

The Importance of Being EDUCATED

by Evan Woolley



Bot Building.

Bot Contest.

Oreo Stacking.

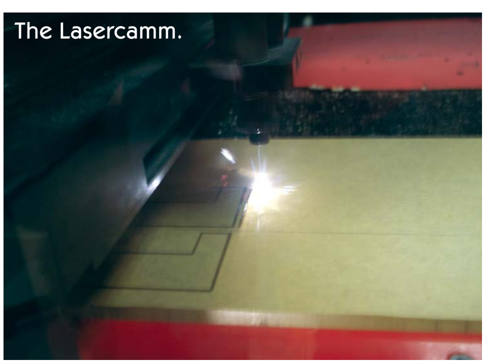
The University of California, San Diego — a part of the prestigious University of California system — is certainly a major engineering university in the United States — it was named the university that was "Hottest for Science" in 2006 by *Newsweek*. I am an undergraduate mechanical engineering major at UCSD, and I have had the pleasure of seeing how the university that is "Hottest for Science" works from the perspective of a student.

Even after only one year I know that I have learned a great deal that will help me in an engineering career, and perhaps my most helpful class so far was MAE (Mechanical and Aerospace Engineering) 3, Introduction to Engineering Graphics and Design.

MAE 3 introduces students to engineering graphics techniques with AutoCAD and design with hands-on projects — namely a mechanical clock and a robot contest. The MAE 3 Robot Contest provides an important glimpse into the academic side of engineering, which is significant for both students who want to get an idea of what they're getting themselves into if they pursue an engineering degree and employers that want to know how possible future employees are being educated at the university level.

Does Everybody Know What Time It Is?

As the course title implies, MAE 3 encompasses two major aspects of engineering: graphics and design. The graphics half of the class — taught by Professor Bill Bussard — teaches students how to do hand sketching, two dimensional AutoCAD, and three dimensional Autodesk. The design half of the class teaches proficiency in various fabrication tools like drill presses, band saws, and the super cool Lasercamm.



The Lasercamm.

MAE 3 is all about gaining practical experience. The students learn how to express and communicate their ideas with computer software used in real-world engineering, and they gain invaluable shop skills and see that engineering is not all about physics formulas and free body analysis — it's also about getting your hands dirty working with machine tools and metal.

The MAE 3 course is divided into two parts: lecture and lab. During lecture, the professors speak on topics ranging from dimensioning and tolerancing for AutoCAD and creativity, teamwork, and engineering analysis for design. The laboratory section is led by undergraduate tutors, and this is

where the building takes place.

The actual "lab" is the Design Studio, a lab equipped with drill presses, metal shears, a band saw, a Lasercamm, and much more. The Design Studio was made in such a way to accommodate both the students coming in with some design experience and the students that would be working with a drill press for the first time.

Clock Time

The MAE 3 course is characterized by two projects: a mechanical clock and a competitive robot. Both serve to emphasize the theoretical and practical aspects of engineering projects. The clock is primarily an introduction to engineering analysis and the tools in the Design Studio; the robot contest is where the real action is. The balance the class tries to strike is one between practical trial and error and academic analysis. Students in MAE 3 get their first taste of that with the mechanical clock project.

The mechanical clock is a simple machine that is essentially made up of an escapement wheel and a pendulum. Both of these major parts were designed on AutoCAD and then cut out on the Lasercamm. The Lasercamm is a $100,000 machine that uses a laser beam to cut out two dimensional drawings from programs like AutoCAD onto materials such as acrylic. It's definitely a cool tool.

The rest of the parts for the clock (bearings, shafts, spacers) were all precut, so the rest of the project was basically assembly. Once the clocks were built, the students performed two types of analysis on them: a point mass analysis and a rigid body analysis. The point mass analysis assumed that all of the mass of the pendulum was concentrated at a single point, while the rigid body analysis took into account the irregular geometry of the pendulum.

Basically, these two analyses showed that a theoretical analysis can effectively approximate actual performance (in this case, the timing of the clock), and that a more specific and sophisticated analysis meant great dividends for accuracy (the rigid body analysis was much more accurate than the point mass analysis).

Robot Teams Unite

The year at UCSD is divided into 10-week quarters, and the clock project really only took up the first 2.5 weeks of the class. The rest of the entire class was dedicated to the robot contest. Just like real-world engineering projects, the MAE 3 students tackle the robot contest in teams. My team consisted of myself, Monique Ochoa, Peter Nguyen, and Wai Keung (Louis) Lee. The class emphasized the importance of teamwork — one lecture was even entirely devoted to the subject. The commitment to teamwork extended to the lab section as well, where the first team activity for every group of students was to build the tallest tower possible out of a finite number of Oreos. Indeed, after fumbling around with sticky Oreos and brainstorming about what type of building blocks we could make out of the cookies, we did feel more like a team.

Stacking Rings

After being put into teams, the class was introduced to the 2006 contest — Ring Stacking. Inspired by the child's toy stacking rings, robots would have to manipulate various sizes of rings and move them from post to post. The contest table had 11 posts. Three posts on each side held inverted stacks of rings. Two other posts on each side that were empty at the start of the competition were goal posts — depositing a ring on one of these would score one point.

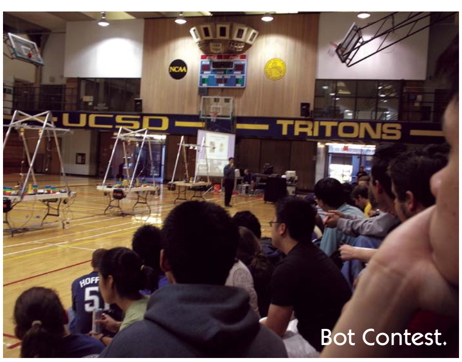
At the center of the field was a conical post. Rings placed on this post would score two points, but if you wanted to score more than three rings on the center post, the stack would have to be inverted somehow because of the posts' conical shape.

The competition was head-to-head, and whichever team had the most points at the end of one minute won the match. The robots themselves had to fit inside a 10" x 10" x 10" box, and they could only be constructed out of the materials included in the kit of parts.

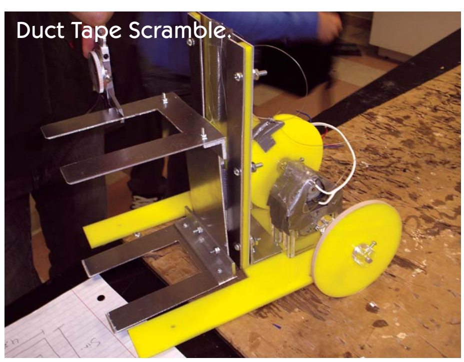
The kit of parts given to each team was simple. As far as conventional power sources, the kit had three non-geared motors and two geared motors. Also included was an assortment of rubber bands, springs, and a solenoid. As far as structural material, the kit contained various thicknesses of acrylic sheets, two sheets of aluminum, and a few pieces of aluminum angle.

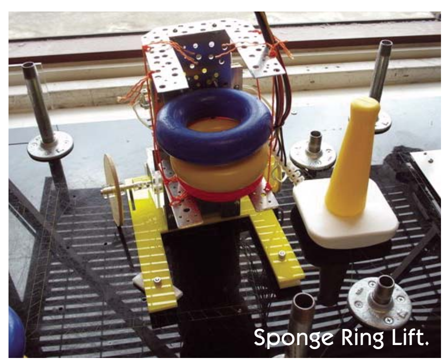
The last miscellaneous bits in the kit included aluminum shafts of several sizes, super glue, o-rings, and allotments for a certain length of string and wire. The only fasteners that teams were allowed to use were the ones in the Design Studio, which amounted to various lengths of 4-40, 6-32, 8-32, and 10-32 screws, washers, and hex nuts. There were also two different metric sized screws for the non-geared and geared motors.

The first task for the team was concept generation. No idea was too wacky or off-the-wall. After some brainstorm-



Bot Contest.


Duct Tape Scramble.


Sponge Ring Lift.

ing outside of the box, my team had ideas as diverse as giant rotating arms, flippers, forklifts, and defensive nets.

Risky Business

The class stressed a very systematic approach to the robot project, which included practices like risk reduction and energy analysis. After initial designs were conceived during team concept generation, the best ideas were determined using a Pugh Chart that ranked the various designs on criteria from ease of manufacturability to point possibility to coolness.

After examining our ideas, the team began to lean toward a forklift arm with an "inversion mechanism" all mounted on a mobile base. The simple up and down forklift claw would grab the rings and lift them off of the posts, and the "inversion mechanism" would turn just the forklift claw upside down, inverting the ring stack so it could be scored on the center post.

Once the best design was distilled, risk reduction was undertaken. The highest risk elements of the design were mocked up using foam core and tested to determine if the designs were indeed feasible. My team mocked up the forklift claw and a rough approximation of the inversion mechanism. We decided one of the highest risk items in our simple design had to do with size. We weren't sure if the inversion mechanism would be able to be made small enough so that it would fit reasonably behind the forklift claw.

After mocking up the forklift claw with the maximum dimensions it would possibly need and also attaching a rudimentary set of foam core wheels to approximate any type of gear train or pulley system we might use for the power transmission, we could see qualitatively that the inversion mechanism would not be infeasible because of size. Other risk reduction tests done by other teams included comparisons of two different ideas for end effectors, and a quick and dirty energy or torque analysis to see if an ambitious arm design could actually be built.

In any case, the risk reduction tests were a valuable part of the design process. For my team, they confirmed the feasibility of our design. Other teams realized that their ideas needed serious revision, and yet others found that maybe a combination of two distinct ideas they had could actually be the best solution.

Overall, the risk reduction tests were ways to take a step back and size up the favored ideas for the robot designs; ways to discover the most effective ideas before blindly diving in and making a lot of progress on a design before figuring out somewhere down the road that maybe it wasn't such a good idea after all. It prevented students from working on a design and using up material only to later come to the unwanted realization that "it was a good idea at the time."

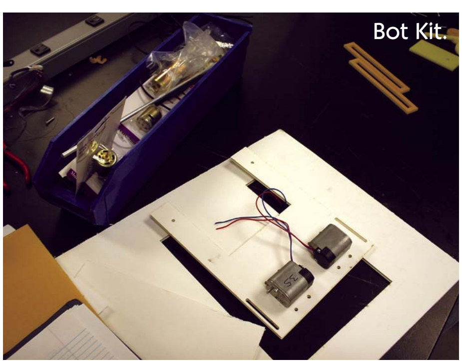
The most sophisticated risk reduction tests took into account some form of energy analysis. The kit of parts was limited in the sense that it only had two types of motors — one geared and one non-geared — so teams could not safely work under the assumption that they could no matter what make any idea they come up with work — in some cases, there simply might not be enough energy in the kit to perform a certain action.

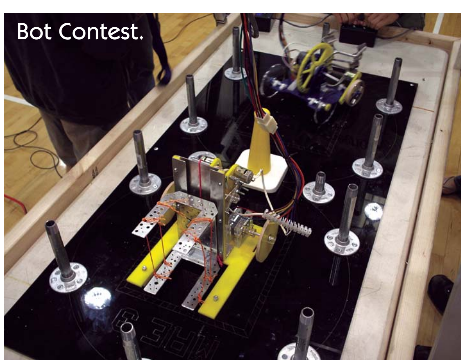
My team's design was basically three mechanisms: the forklift, the drive train, and the inversion mechanism. We intuitively wanted to use the geared motors for the drive train, because they would easily have the requisite energy and their slower speed would be better for a more controllable robot. That left the forklift and the inversion mechanism. If the inversion mechanism was centered on the forklift claw, rotating the claw to invert the stack would not take much energy at all, so our main concern was with the forklift.

The mechanism we wanted to use for the forklift was some kind of drum that would wind up a string that was attached to the forklift. As the string was wound up, the forklift would lift. Since we were using our geared motors for the drive train, we were only left with the non-geared motors for the forklift. A reduction was obviously necessary, and some energy analysis would certainly be needed to determine things like the gear reduction needed. But that would have to come later, because we had some deadlines to meet.

Long Lines and Deadlines

After each team presented their risk reduction tests to the class, it was time to set our sights on the next major deadline — the moving part. My team ambitiously set the goal of finishing the drive train by that deadline, so the moving base of our robot would be our moving part. All we had to do was make motor mounts for the geared motors, wheels, the robot floor, couplings, and shafts. Most of these parts had to be designed on AutoCAD then cut out on the Lasercamm, which doesn't seem like


Bot Kit.


Bot Contest.


Friction Drive.

a big problem. The problem, however, is a class of about 200 students and one Lasercamm. Each team only had very limited time on the Lasercamm, and my team was able to get everything cut out but our floor. With that being the case, we weren't able to assemble the complete drive train, but a complete assembly of a direct drive between the geared motor and a wheel was enough to meet this deadline.

The next major deadline was to demonstrate a machine that could score one point. After cutting out the floor and finishing the drive train, the team decided that all we needed to score one point was a forklift that could go up and down; the inversion mechanism could be dealt with later.

the aluminum sheet on the band saw, and so was the back plate and front flanges for the upright. The forklift claw was designed symmetrically for ease of manufacturability and for practical reasons — if the inversion mechanism was used, then we would want it to be able to lift the ring stacks with the claw right side up and upside down.

The idea behind the upright was to make it as low friction as possible by using aluminum. The only pieces we needed cut by the Lasercamm for these components were some spacers between the back plate and flanges for the upright. We could have cut these with the band saw, but the Lasercamm yielded a cleaner edge and there was no line for the machine at the time.

better idea to do the analysis first, but we were rushed. After trying to score a point with the floundering forklift that didn't lift, though, we were motivated to find out what went wrong and how to fix it. So we busted out our calculators.

First, we calculated the tension required to lift the forklift:

$T = m_{claw} * g$
$T = (0.396\ kg) * (9.8\ m/s^2)$
$T = 3.88\ N$

Then we calculated the tension supplied by our direct drive assembly:

$T = \tau_{motor} / r_{innerdrum}$
$T = (0.0140\ Nm) / (0.0254\ m)$
$T = 0.2756\ N$

> **"MAE 3 introduces students to engineering graphics techniques with AutoCAD and design with hands-on projects — namely a mechanical clock and a robot contest."**

Floundering Forklifts and Torque Tribulations

The basic idea for our forklift was that a non-geared motor would drive a drum to wind up a string attached to the forklift. The major components we needed to build were the forklift claw itself, the upright, and the power train. The claw and upright were easy pieces to start with, since they didn't need a lot of pieces cut out by the Lasercamm. After cutting it close on the moving part deadline, we decided that the fewer parts that needed the Lasercamm, the better.

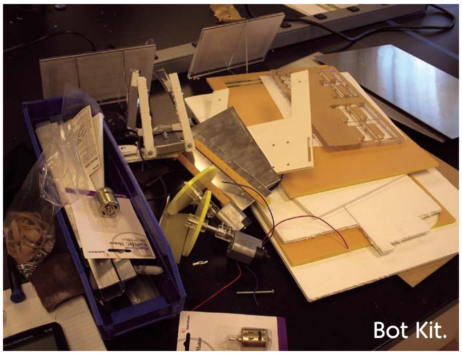
So the forklift claw was cut out of

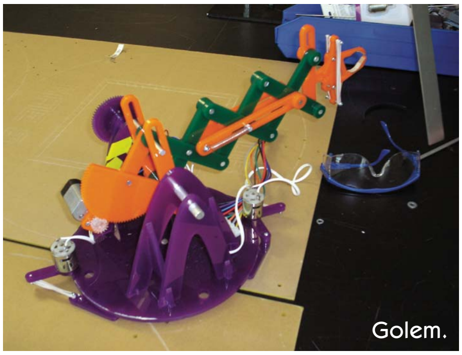
After attaching the upright to the base with an angle bracket and attaching the claw to the upright with a rudimentary back plate, we were ready to concern ourselves with the power transmission. Unfortunately, we were running out of time before the score one point deadline, so we just bodged together a direct drive for the non-geared motor to the forklift drum. Since you can't just feed tuna fish mayonnaise, we should have expected that a direct drive on the non-geared motor would be unsuccessful, so we had a major redesign on our hands. It was time for some analysis.

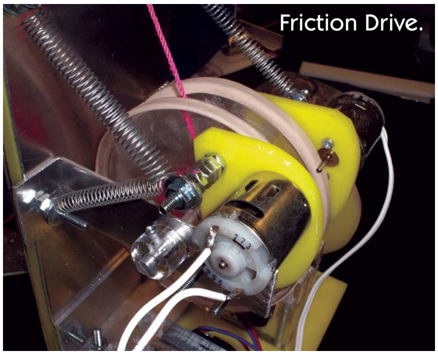
It probably would have been a

Obviously, we had a problem. To remedy our tension problems, we went back to what was actually the original design for the forklift power transmission before we were rushed — a friction drive. The friction drive would be simple — no gears and no pulleys. And a friction drive could easily achieve a large reduction in a small space, because the input radius would be the radius of the motor shaft — only 0.09 inches.

For our new design, we chose a four-inch diameter wheel for the friction drive. The number was chosen as one of the biggest size wheels we could fit on the robot, and we figured


Bot Kit.


Golem.


Friction Drive.

that a 44:1 reduction would certainly provide the requisite torque and tension. We also reduced the size of the inner drum that the string would wrap around.

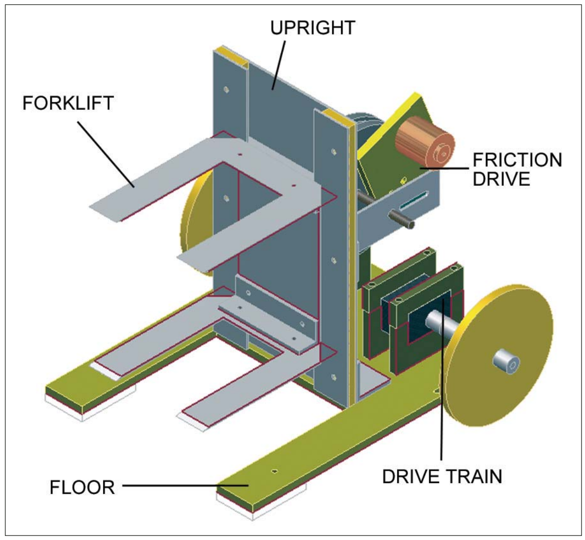
In the end, the theoretical tension supplied by our friction drive was around 40 N, so we certainly had enough to lift our forklift. For the sake of symmetry, we added a second non-geared motor to the other side of the friction wheel, and we also thought it would be a good idea to add a second one because of the frictional losses that we did not take into account in our analysis.

The Big Game

By the time we sorted out our forklift, the end of the quarter was upon us, so the inversion mechanism had to fall by the wayside. Even so, our forklift proved to be a reliable way to score points, so we had a competitive robot. Unfortunately, we did miss a deadline to compete against the Golem — a robot that the class tutors put together. Nevertheless, our robot was done on time for the competition, so we were all happy about that.

Before the teams in the class had a chance to go head-to-head, every team had to make an oral presentation on their robot. Being able to clearly communicate your ideas is a trademark of good engineering, so it really makes sense that students would have to practice that at the university level. After my team, Spongebob (the name came from Peter's T-shirt that he was wearing when our tutor asked us what our team name was; I also tell myself that it's appropriate because the forklift claw looked a bit sponge-like after we drilled a bunch of holes in it to lighten the load) made our presentation about things like the evolution of our forklift design, we were ready for competition.

The competition was two parts: a section competition where teams competed with everyone from their lab section and a class wide single elimination tournament where all 50 or so teams competed. Team Spongebob scored points in every round, but we did not win. That's okay, though, because just seeing the robot work after seven weeks of work was well worth it.



So What Did You Learn in School Today?

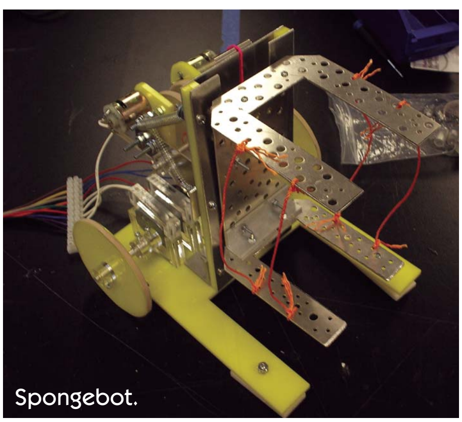
As a FIRST alumni and a veteran of various other robotics competitions, I am compelled to compare my experience in MAE 3 with my experiences in other robot contests.

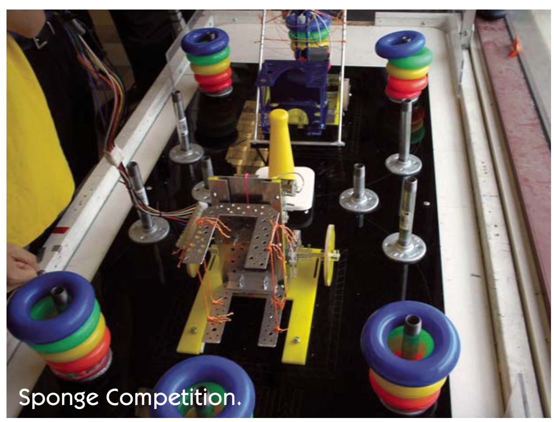
Compared to things like FIRST, it seems, well, smaller. This is certainly understandable given that MAE 3 is a single university course while FIRST is an international competition, but I couldn't help feeling that while I was sitting in the gym awaiting the final contest, staring up at the tiny competition tables, that it was so much smaller than FIRST. But then I reflected on the MAE 3 build, and I felt like it was certainly similarly stressful and a comparable mental workout to the six weeks of craziness that characterize FIRST.

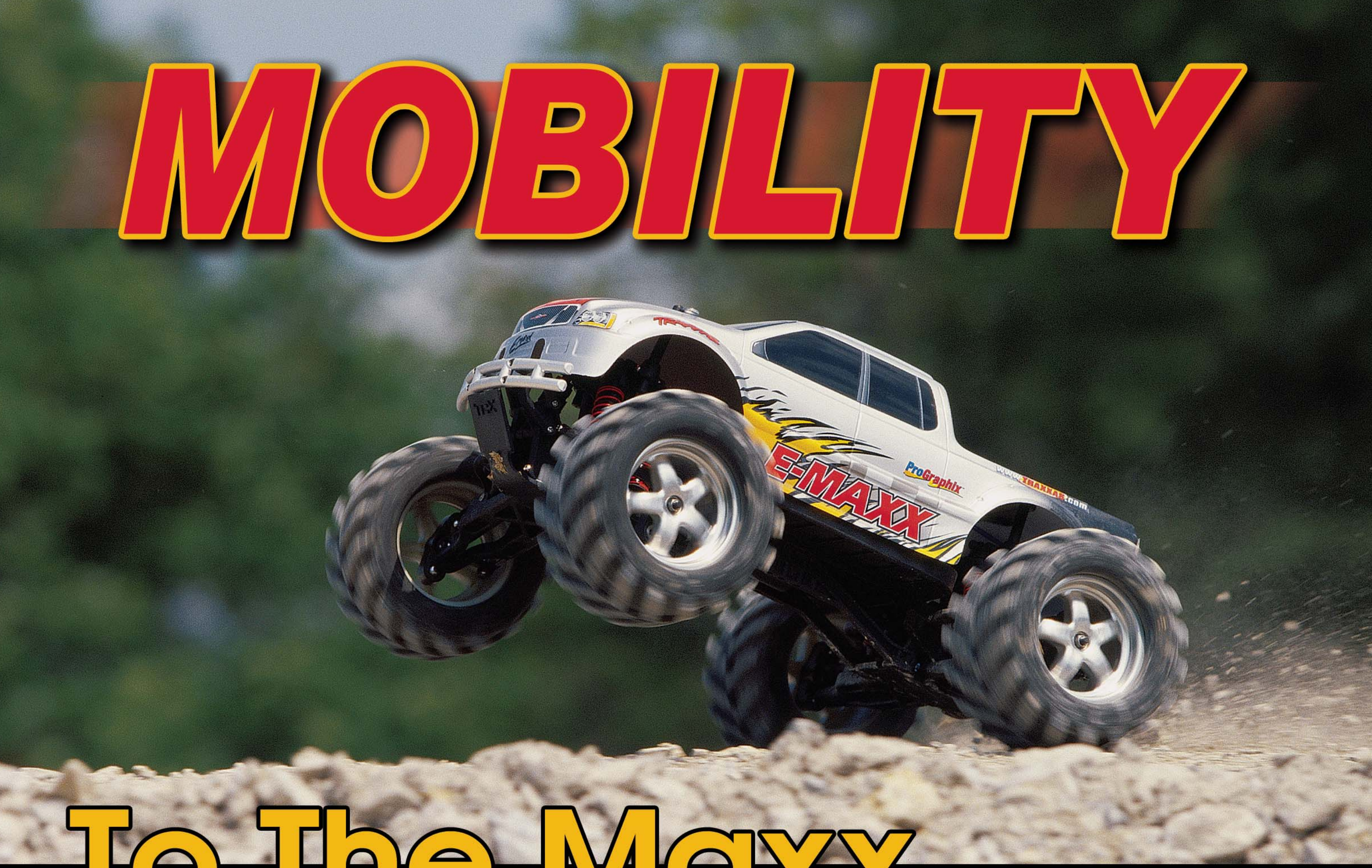
There is also a comparable diversity of designs in both competitions. At the MAE 3 robot contest, I saw a few other forklifts, but I also saw rotating arms, stationary claws, scissor lifts, and anything else that a bunch of creative college students can come up with in just a few weeks. Then I thought about how I had to whip out my calculator a lot more during the MAE 3 course.

Sure, FIRST involves the same physics, but it doesn't really have a set curriculum. The methods that teams use to build their robots in FIRST are as varied as the robots themselves. I know from my experience with Team 1079 that we didn't focus that much on analysis. We were the type of team that liked to do trial and error, the type that would sit in a restaurant volleying ideas back and forth with nothing more than an intuitive sense that "this should work" or "this one is unrealistic."

That may work with a high school competition, but real engineering stresses a balance between analysis and trial and error. That is a balance that the MAE 3 class achieved. It introduced engineering majors early on to the shared importance of number crunching, risk reduction, and good old bodging. SV


Spongebot.


Sponge Competition.



To The Maxx

PART 5 — Right to the Point

→ by Chris Cooper

For more information on this product, visit **www.machinebus.com/emaxx**

Last month, we demonstrated the effectiveness of using dead reckoning navigation over short distances. Using a newly added digital compass for heading updates, along with odometry estimates based on the encoder readings, we were able to navigate around various short courses and successfully arrive back at the starting point.

In this article, we'll add to our navigation techniques and get right to the point — GPS point, that is. We implement GPS navigation by installing a GPS receiver and then follow a trail of GPS waypoints to reach the final destination, just like the competitors in the DARPA Grand Challenge.

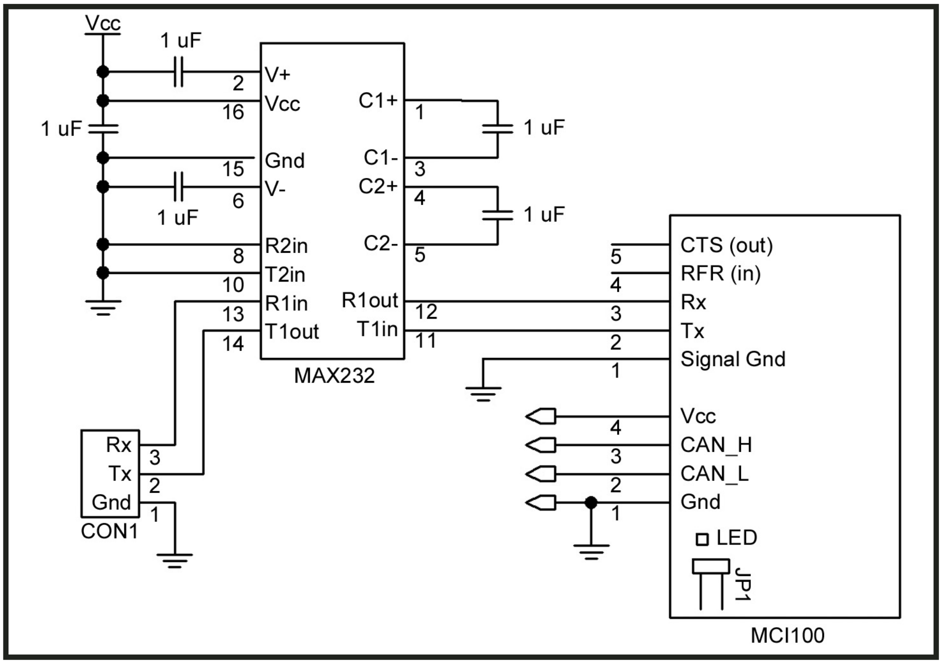
GPS Navigation

GPS navigation is a navigation technique that is complimentary to dead reckoning navigation. It is useful when you know the longitude and latitude of your destination and any points in between. Since GPS readings are not subject to the cumulative errors that are inherent in dead reckoning (see last month's article), it's a great way to accurately navigate over longer distances.

The first satellite for the Global Positioning System was launched in 1978, and by 1994, a complete constellation of 24 satellites was attained. Interestingly, each satellite is built to last about 10 years and replacements are constantly under construction. Each satellite transmits two low power (50 W) line-of-sight signals that can pass through clouds, glass, and plastic, but cannot penetrate buildings or even dense foliage. The signal contains the identification

Photo Above: The E-Maxx RC monster truck makes an excellent robotics base. *Photo courtesy of Traxxas.*

Figure 1. Schematic showing wiring of the Garmin GPS V through CON1 to the MCI-110 communication device.



of the satellite, where each satellite should be at any time throughout the day, and the current date and time which are all essential for establishing position.

GPS receivers can determine their current position by measuring the time delay between when the satellites sent the signal and when the signals are received. At least three satellites need to be acquired to determine position, four if you need to know your altitude. The more satellites acquired, the more accurate your position information will be.

As for accuracy, a standard GPS receiver can be accurate to within three meters. However, this accuracy is degraded to about 15 meters due to error from sources such as atmospheric effects and satellite ephemeris and positioning errors. Using techniques such as Differential GPS or WAAS (Wide Area Augmentation System), accuracy can be improved to between 1-3 meters. If you need higher accuracy than that, there are subscription services like the Starfire Network or dual frequency receivers like the Trimble BD950 that are accurate to the decimeter.

Adding a GPS Receiver

The GPS receiver I've chosen to use is a Garmin GPS V, but any GPS receiver with serial output will work. The GPS V is WAAS-enabled with routing capability and an RS-232 NMEA 0183 interface. It also has an indoor mode that simulates travel along a route, which makes it ideal for development and debugging for those of us who happen to live and work indoors. Since the serial output from the Garmin is at RS-232 voltage levels, we'll need to add an RS-232 to TTL converter and some additional circuitry to get the GPS V to talk to the MCI-100 at TTL voltage levels.

I've found that connecting the GPS V serial connector to the breadboard with a smaller custom cable is a lot cleaner than using the large serial cable that came with the GPS. Another option would be to purchase an additional cable and cut that down to a more manageable size.

Processing GPS Output

NMEA Sentences

GPS communication uses the protocol defined by the National Marine Electronics Association (NMEA) specification. The protocol involves lines of data called 'sentences' which are independent of each other. All sentences begin with a '$' followed with a two-letter prefix that defines the device type. Sentences end with a '*' followed by a checksum and a carriage return/line feed. The data within each sentence is ASCII text separated by commas. Standard GPS sentences have a two-letter prefix of 'GP' and device manufacturers may also define

TABLE 1. GPGGA — Fix Information

The GPGGA message provides fix data and accuracy.

$GPGGA,155334,4202.5159,N,08742.5610,W,8,05,2.0,221.5,M,-34.3,M,,*74

155334	Fix taken at 15:53:34 UTC
4202.51, N	Latitude taken at 42 degrees 02.61' N
08742.5610, W	Longitude taken at 087 degrees 42.5610' W
8	Fix quality: 0 = invalid 1 = GPS fix (SPS) 2 = DGPS fix 3 = PPS fix 4 = Real Time Kinematic 5 = Float RTK 6 = Estimated (dead reckoning) (2.3 feature) 7 = Manual input mode 8 = Simulation mode
05	Number of satellites being tracked
2.0	Horizontal dilution of position
221.5, M	Altitude above sea level in meters
-34.3, M	Height of geoid in meters
*74	Checksum

TABLE 2. GPRMB — Recommended Navigation Data

The recommended minimum navigation sentence is sent whenever a route is active.

$GPRMB,V,0.00,R,WP1,WP2,4202.5159,N,08742.5610,W,0.000,270.0,,V,S*31

V	Data status A = Active, V = Void (warning)
0.00	Cross-track error (nautical miles, 9.99 max)
R	Steer right to correct (or L = left)
WP1	Origin waypoint ID
WP2	Destination waypoint ID
4202.5159,N	Destination waypoint latitude
08742.5610,W	Destination waypoint longitude
0.000	Range to destination, nautical miles (999.9 max)
270.0	True bearing to destination
	Velocity towards destination, knots
V	Arrival alarm A = Arrived, V = Not Arrived
*31	Checksum

TABLE 3. GPRMC — Recommended Minimum Position, Velocity, and Time Data

Shows the current latitude, longitude, velocity, heading, and magnetic deviation.

$GPRMC,155334,A,4202.5159,N,08742.5610,W,0.0,0.0,080606,3.1,W,S*16

155334	Fix taken at 15:53:34 UTC
A	Data status A = Active, V = Void (warning)
4202.5159,N	Current latitude
08742.5610,W	Current longitude
0.0	Velocity (in knots)
12.4	Heading in degrees true
080606	Date 08/06/2006
3.1,W	Magnetic deviation
S*16	Checksum

proprietary sentences, as well. Garmin, for example, uses a prefix of 'PG.'

If you have a GPS with serial output, it's easy to connect it to your computer to see what NMEA sentences are being output. Simply connect the GPS receiver to your computer's serial or USB port and open that port with a terminal program such as Kermit. Typically, the output baud rate on GPS receivers is at least 4800, which allows the same set of sentences to be output about every two seconds.

Useful Standard NMEA Sentences

E-Maxx GPS navigation can be accomplished by processing three standard NMEA sentences: GPGGA, GPRMB, and GPRMC. I've included some sample sentences and field descriptions, which are shown in Tables 1, 2, and 3.

GPS Code

The NMEA sentences mentioned above are all that's needed for GPS navigation along a route. Each sentence is processed as a stream on the MCI-100. As soon as a field is read in, the information is extracted, compressed, and sent over the CAN bus in real time before the complete sentence is finished being read. This approach reduces the memory footprint on the MCI-100 by only keeping the minimum amount of sentence content around.

On our application host, the interface to access the information looks like Listing 1.

Setting a Course — Routes and Waypoints

Now that the application host has

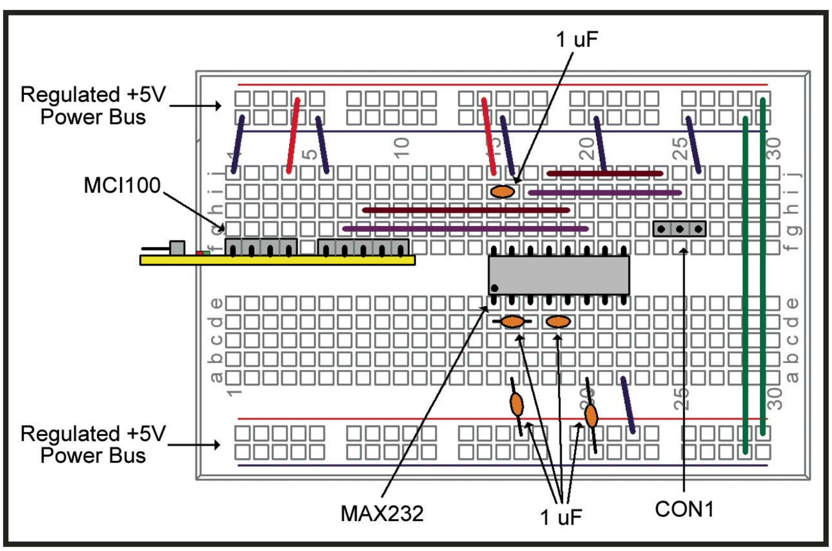


Figure 3. GPS enabled E-Maxx.

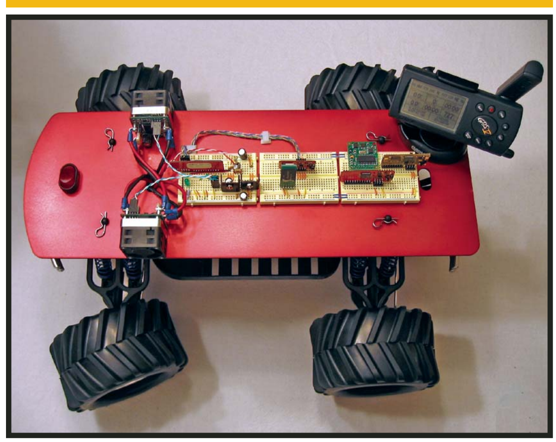


Figure 2. Diagram of GPS navigation module.

the latest GPS information readily available, our next task is to set up a route for testing. A route is an ordered set of waypoints leading towards a destination. Each waypoint is specified with a longitude and latitude. In order to traverse a route, the E-Maxx navigates to the nearest waypoint, and then to the next one in turn, until the destination waypoint is reached.

The Garmin GPS V receiver can record a waypoint at its current location to an accuracy of within three meters or better if using WAAS capability. Waypoints can also be entered manually on the receiver as a pair of coordinates, or marked on a "compatible" computer mapping program and uploaded to the GPS receiver.

Over the years, a myriad of vendor specific formats have been created to hold waypoint, track, and route information for use in mapping programs and for importing and exporting into and from GPS receivers. GPX is an open standard based XML format created to address this problem. Additionally, a very useful cross-platform application called GPSBabel can convert between GPX and numerous other formats. It can also communicate with numerous GPS receivers, including the Garmin GPS V, to upload/download routes. More information can be found online by following the links at the end of the article.

Following a Course — Navigation Code

Now that the route has been uploaded to the GPS receiver using GPSBabel, we can update the navigation code on the application host to follow waypoints. The following code excerpt shows how driving from waypoint to waypoint is accomplished.

gps_getMagneticBearingToNextWaypoint() internally compensates for the magnetic declination (the angle between magnetic north and true north) at our current position. We can then pass the magnetic bearing along with the compass reading directly into the steering control.

As we get closer to the waypoint, inaccuracies in the receiver will cause the heading to fluctuate. In order to filter out errors and fluctuations in the readings, we calculate a moving average for the heading. The moving average gets reset each time we

PARTS LIST

- ⇛ MCI-100 (**www.machinebus.com**)
- ⇛ DS-232 TTL-RS232 shifter (**www.jameco.com**)
- ⇛ Any GPS with serial output (I used a Garmin GPS V)

LISTING 1

```
#include <stdlib.h>
#include <unistd.h>  /* UNIX standard function definitions */
#include "machineBus.h"
#include "nmea.h"
#ifndef GPS_
#define GPS_

    typedef struct
    {
        LOCATION    latitude;
        LOCATION    longitude;
    } FIX;

    struct gps_t;
    typedef struct gps_t *Gps;

    // Create a new gps reference
    Gps gps_createGps(CommBus C);

    Gps gps_createGpsDebug(CommBus C);

    // Get the number of satellites - GPGGA
    uint8_t gps_getNumberOfSatellites(Gps G);

    // Get the horizontal dilution of precision
    float gps_getHorizontalDilutionOfPrecision(Gps G);

    // Get the current location
    FIX gps_getCurrentFix(Gps G);

    // Get the route data status
    uint8_t routeDataStatusOk(Gps G);

    // Get the next waypoint fix
    FIX gps_getNextWaypointFix(Gps G);

    // Get range to next waypoint in meters
    float gps_getRangeToNextWaypoint(Gps G);

    // Get magnetic deviation
    float gps_getMagneticDeviation(Gps G);

    // Get magnetic bearing to next waypoint
    float gps_getMagneticBearingToNextWaypoint(Gps G);

    // Get true bearing to next waypoint
    float gps_getTrueBearingToNextWaypoint(Gps G);

    // Arrived at waypoint
    uint8_t gps_getArrivedAtWaypoint(Gps G);

    // Dispose of the gps
    void gps_disposeGps(Gps G);

    uint8_t gps_messageCallback(void* object, CAN_MESSAGE *rxMessage);

#endif /*GPS_*/
```

LISTING 2

```
done = 0;

while (!done ) {

    /* Check Bus status */
    if (commbus_status(bus) != 0) {
        printf( "Failed to retreive status\n" );
    }
    // If the range to the last waypoint is less than dilution of precision (error),
    // and we are at the end waypoint
    if ((gps_getRangeToNextWaypoint(gps) < gps_getHorizontalDilutionOfPrecision(gps))
            && strcmp(gps_getNameOfNextWaypoint(gps), "END") == 0) {

        done = 1;

    } else if ( commbus_readyToTransmit(bus)) {

        if (gps_arrivedAtWaypoint(gps)) {
            // New bearing, reset moving average
            util_resetMovingAverage(movingAverage);
        }

        // Use a moving average to account for noise/errors
        if (gps_routeDataStatusOK(gps)  {
            gpsBearing = util_adjustMovingAverage(movingAverage,
                                 gps_getMagneticBearingToNextWaypoint(gps));
        }
        // Get our current bearing
        currentBearing = compass_getBearing(compass);
        steering_steer(steering, gpsBearing, currentBearing);
    }

} // while
```

arrive at the next waypoint, in preparation for the new heading on the next leg of the route. We determine the end of the route by specifically naming the last waypoint "END" and stopping when we are within the current accuracy limitations of the GPS receiver.

RESOURCES

⇛ Autonomous E-Maxx:
www.machinebus.com/emaxx

⇛ Garmin:
www.garmin.com

⇛ GPX – GPS Exchange Format:
www.topografix.com/gpx.asp

⇛ GPSBabel:
www.gpsbabel.org

⇛ Maxim IC RS-232 Transceiver:
www.maxim-ic.com/Interface.cfm

Conclusion

After adding the GPS receiver and modifying the code to follow a route of GPS waypoints, we can now traverse relatively large distances with a few caveats. The current code doesn't handle the loss of the GPS signal well. If we lose the signal, the E-Maxx will continue along the current heading until it picks up the signal again or until it runs into something.

During our trials, we've insured the routes traveled are clear and have therefore "avoided" the obstacle problem. In the next article, we will deal with collision avoidance and detection by adding a variety of ranging sensors. We will then be able to detect objects on our route and avoid collisions, then get back on track towards the next waypoint. **SV**

AUTHOR BIO

Chris Cooper is currently a software architect for Chicago-based Machine Bus Corporation. He has a B.S. in Computer Science from the University of Illinois, has presented at the OMG's Robotics SIG on Distributed Control Systems, and is a member of the Chicago Area Robotics Group (Chibots). He can be reached at cooper@coopertechnical.com

Building (H-)Bridges

— Part 2 —

by Peter Best

Let's add some intelligence to the H-Bridge hardware I introduced to you last time. For those of you that are just joining us, the H-Bridge hardware that was previously presented in SERVO is shown in Photo 1.

To recap for those of you that missed the introductory *SERVO* H-Bridge column, we took some off-the-shelf Microchip MOSFET drivers and fashioned them along with some glue logic to act as driver circuitry for a quad of dual-MOSFET devices. After all of the wire slinging was said and done, we ended up with a pair of H-Bridges suitable for driving small brushed and stepper motors.

When we're finished discussing the schematics, code, and photos in this month's *SERVO* offering, you'll know how to apply the services of Microchip's newest motor control oriented PIC — the PIC16HV616 — to the H-Bridge circuitry you see in Photo 1. With that, let's begin the H-Bridge driver design cycle.

The PIC16HV616

The 14-pin PDIP version of the PIC16HV616 can be seen dominating the intelligent H-Bridge driver componentry shown in Photo 2. The PIC16HV616 speaks standard PICese using only 35 assembler instructions. A precision internal oscillator provides either a 4 MHz or 8 MHz system clock for the PIC16HV616's integral subsystems. A standard crystal or ceramic oscillator can also be attached to the PIC16HV616 to provide higher or lower clock rates.

The PIC16HV616 is actually a derivative of the PIC16F616. The PIC16F616 operates over a voltage range of 2.0 V to 5.5 V. Although not stated directly in any of the PIC16HV616 datasheets, the "HV" more than likely stands for High Voltage and this is what makes the PIC16HV616 different.

The PIC16HV616's operating voltage range spans from 2.0 V to a user defined maximum. This is accomplished by applying the services of a shunt voltage regulator that is built into the PIC16HV616. The high voltage capability of the PIC16HV616 allows this PIC to be thrown into motor drive circuitry without the need for an extra voltage regulator for the PIC. Otherwise, the PIC16HV616 and PIC16F616 are logically identical.

The PIC16HV616 embodies all of the standard bells and whistles you normally see in any typical PIC. The PIC16HV616 can sleep on command, reset on brown-out conditions, and protect the code embedded in its program memory. The PIC16HV616 is also capable of high sink and source currents on most of its I/O pins. Two analog comparators with built-in, user selectable hysteresis are accompanied by a unique on-chip SR (Set Reset) latch and a programmable on-chip voltage reference.

There's also an eight-channel 10-bit analog-to-digital converter in the PIC16HV616 mix. A trio of PIC16HV616 timers — Timer0, Timer1, and Timer2 — provide a pair of eight-bit timers in Timer0 and Timer2, while Timer1 (which can also be gated with the T1G input) extends to 16 bits of resolution. Each PIC16HV616 timer can be prescaled, with Timer2 having the ability to be both prescaled and postscaled. Motor control usually implies PWM (Pulse Width Modulation). The PIC16HV616's 10-bit PWM subsystem can be configured with one, two, or four

PHOTO 1. If you wish to build up your own version of the H-Bridge shown here, you can get the H-Bridge ExpressPCB layout file from the *SERVO* website (**www.servomagazine.com**). All of the H-Bridge components are off-the-shelf and can be purchased from many of the distributors that advertise in this magazine.

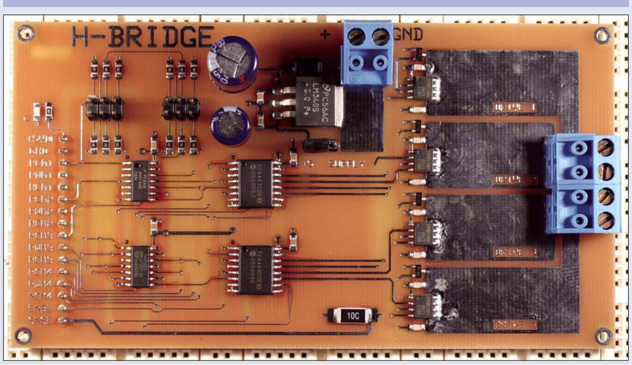
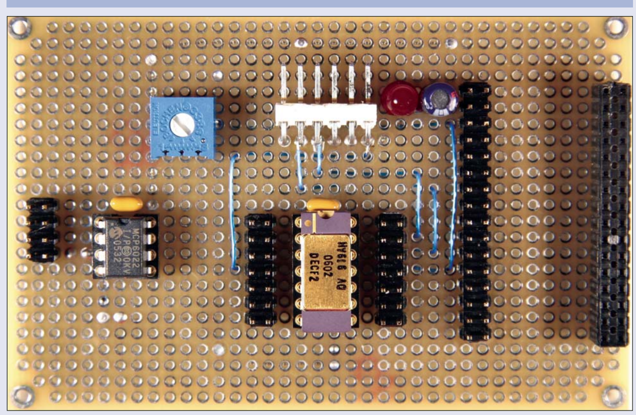


PHOTO 2. The only items that are hard-wired include the ICSP interface, the 10K pot, and the MCP6022 op-amp. Everything that attaches between the PIC16HV616 and the H-Bridge printed circuit board is open game.



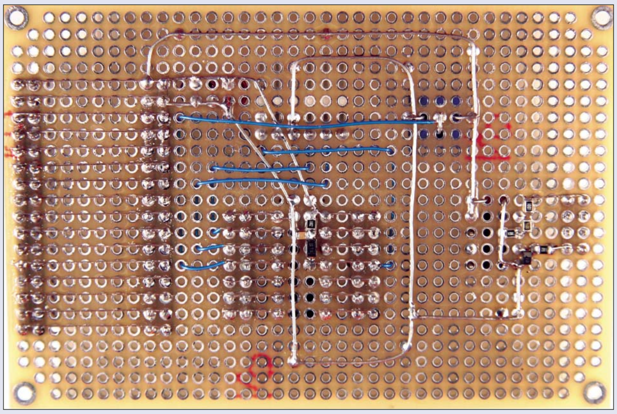


PHOTO 3. This is a shot of my wire art that makes all of the necessary connections between the PIC16HV616, the H-Bridge printed circuit board, and the H-Bridge driver's supporting components. Note my use of SMT passive components to save space and make things neater on the component side of the board.

output channels.

The PIC16HV616's ECCP (Enhanced Capture Compare PWM) module can be programmed to operate in an enhanced mode. The Enhanced PWM Mode can be used to generate a PWM signal on up to four different output pins with up to 10-bits of resolution. This is accomplished using one of four modes: Single PWM, Half-Bridge PWM, Full-Bridge PWM Forward, and Full-Bridge PWM Reverse. We'll use the Full-Bridge modes to drive a brushed DC motor in this installment.

Since the PIC16HV616 is aimed at motor control applications, the program Flash and SRAM complements are small with the PIC16HV616 supporting 2K of program Flash and 128 bytes of SRAM. Although the PIC16HV616 memory numbers may appear to be tiny, remember that many amazing things were done with the original PIC16C5X devices, which had equal or lesser amounts of memory space. In fact, I was told by a Microchip representative that the PIC16C5X parts are still one of Microchip's biggest sellers.

Wire Art

Now that you have the low-down on the PIC16HV616, let's put the part to work. Since there are multitudes of ways to connect to and drive the H-Bridge, the PIC16HV616 H-Bridge driver design must be flexible enough to provide any-to-any connection between the H-Bridge printed circuit board (PCB) and the PIC16HV616 driver board. Take a look at the H-Bridge PCB in Photo 1. There are 16 entry and exit points, which present power, ground, MOSFET driver inputs, an enable input signal, and a sense voltage output for external control and monitoring.

Now take another look at Photo 2. To keep things simple and totally flexible, the H-Bridge driver circuitry is built on a custom high quality perf board, which consists of a .1-inch center set of plated through holes. The use of a perf board implies that instead of a custom PCB, we'll use wire art to fashion our H-Bridge driver circuit.

The H-Bridge's 16-pin interface is populated with a 16-pin single-row header. The H-Bridge driver mates to the H-Bridge 16-pin header with a 32-pin double-row header socket. The header socket could also be a single-row socket but the double row configuration makes it a bit easier to orient vertically and makes for a sturdier mounting point.

Note that the PIC16HV616 is surrounded by a pair of double-row male headers in Photo 2. Also notice that a 32-pin double-row male header parallels the 32-pin double-row female header socket. The method to this header madness is revealed in Photo 3.

Forget about double-row as a physical attribute of the headers as each 32-pin double-row header is effectively a pair of 16-pin single row headers tied together electrically. Another look at Photo 3 shows that each of the row pins on every double-row header is connected to its respective row neighbor via a wire jumper. The 32-pin header and socket on the PIC16HV616 driver board are wired in parallel using wire jumpers forming four columns of 16 pin positions. Each of the 16 rows between the female and male 32-pin headers is connected electrically by a wire jumper.

The same holds true for double-row male headers that surround the PIC16HV616. A wire jumper extends from each PIC16HV616 pin across the pair of adjacent header pins. The double-row pinout arrangement allows us to use temporary wire jumpers with female terminations to connect any point of the H-Bridge 16-pin interface to any pin of the PIC16HV616. In addition, the extra pin in each row is exposed and can be used as an additional jumper point or a test point for monitoring logic levels or voltages.

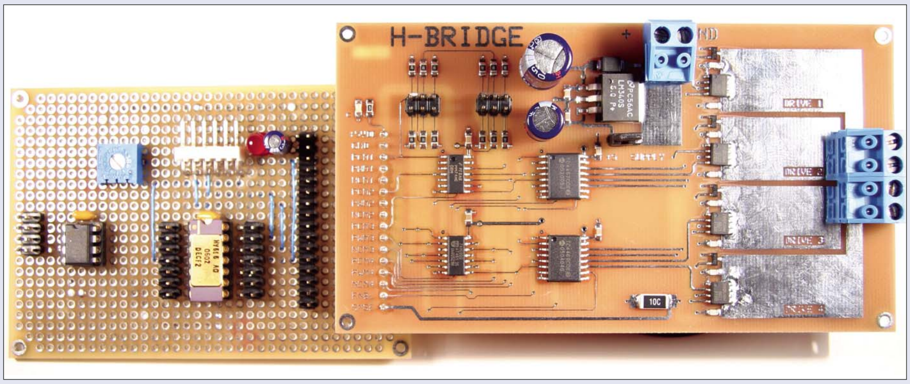
The MCP6022 op-amp is used to amplify the voltage at the H-Bridge's sense output pin. Only the op-amp's input and output pins are accessible via a header pin. All of the passive components associated with the MCP6022 circuit are SMT packages and are mounted under the MCP6022 socket on the wire side of the H-Bridge driver perf board.

I hardwired the 10K pot to the PIC16HV616's AN2 analog-to-digital converter input pin. Since this is all on perf board, you may wish to bring the pot's pins out to header connections for more connection flexibility.

The right-angle six-pin male header shown in Photo 2 is the ICSP programming interface. The PIC16HV616's MCLR reset circuitry is also made up of SMT devices. You can see the reset resistor/capacitor combination and the Vpp blocking diode nestled beneath the PIC16HV616 in Photo 3.

Supply voltage for the PIC16HV616 and the MCP6022 is provided by the H-Bridge PCB regulated power supply via the 16-pin header interface. It is a good engineering practice to include a filter capacitor at the power junction when passing supply voltages between boards using connectors. So, I placed a 10 µF

PHOTO 4. Here's a shot of the H-Bridge and the H-Bridge driver boards mated at the interface point. Wire jumpers with female terminations or wirewrap connections can be used to make connections between the PIC16HV616 and the H-Bridge MOSFET drivers.



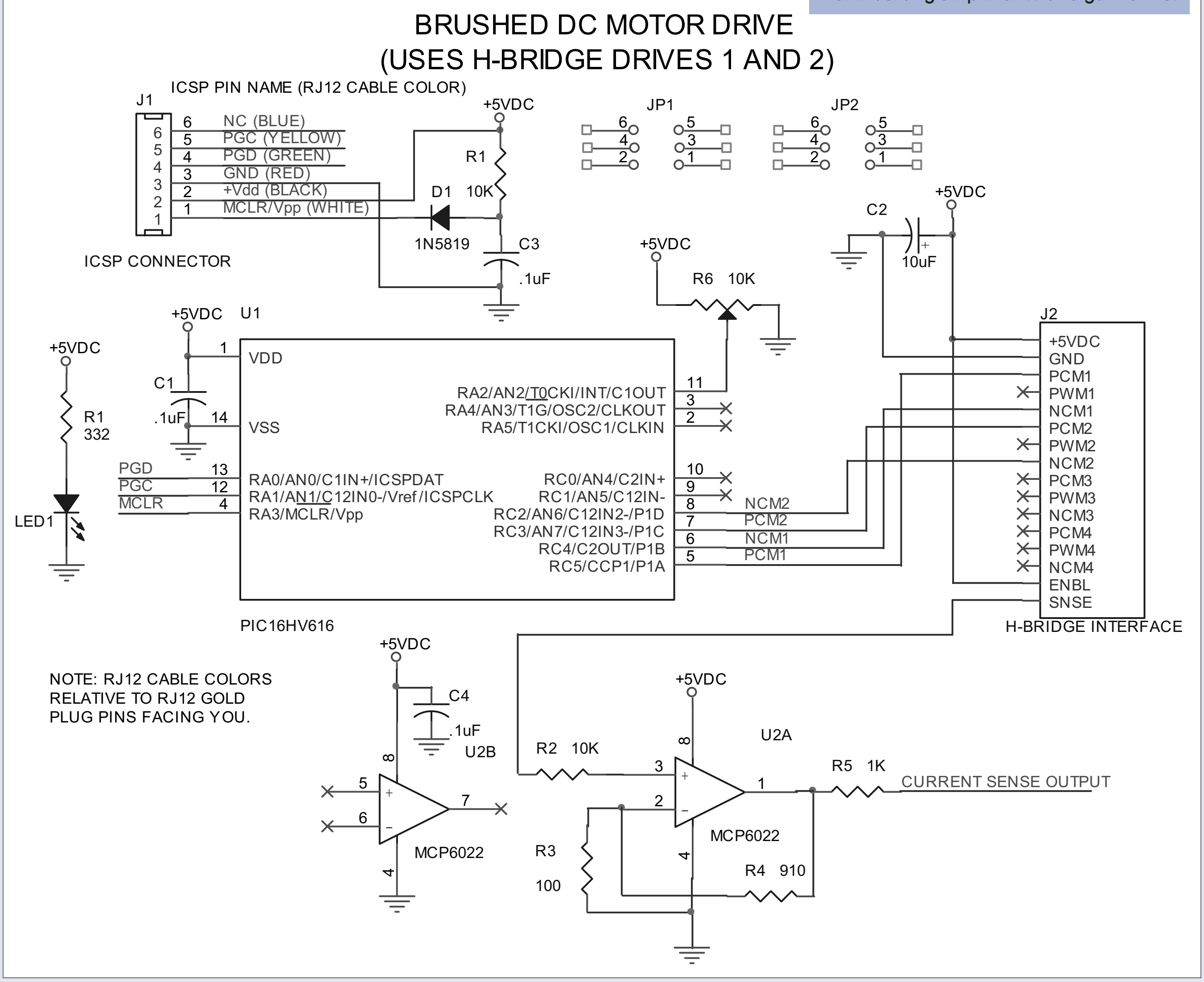
capacitor across the DC supply pins on the H-Bridge driver board.

Just to make sure power was indeed present on the H-Bridge driver pins, I added an LED indicator. Photo 4 is a composite shot of the H-Bridge and H-Bridge driver board mated at their respective 16-position interfaces.

Driving a Brushed DC Motor

With the addition of the PIC16HV616-based H-Bridge driver, the H-Bridge hardware build for our application is complete. Schematic 1 is a graphical depiction of the configuration we will use to attach and drive a brushed DC motor. Note that the H-Bridge segments are not coupled in full-bridge configuration as there are no jumpers on the JP1 and JP2 pins. Only one pair of the H-Bridge's quad of half-bridge drives is needed to drive the brushed motor. The actual motor leads are

SCHEMATIC 1. A five-jumper hookup is all that's needed to drive a brushed DC motor. The MCP6022 is configured as a noninverting amplifier with a gain of 10.



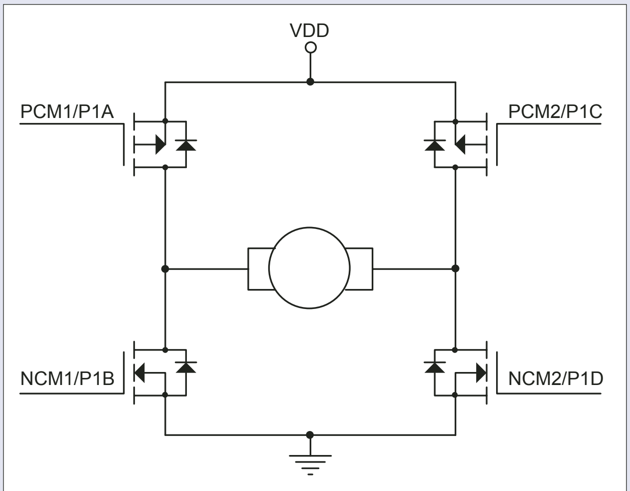


FIGURE 1. The motor leads are attached to each half-bridge segment of the H-Bridge in this manner. Basically, PCM1 and NCM2 are on to drive the motor in one direction and PCM2 and NCM1 work together to drive the motor in the opposite direction. You can get more details by examining the original H-Bridge schematics, which I've provided as part of this month's download package.

connected to half-bridge Drives 1 and 2 of the H-Bridge. A much simplified half-bridge connection diagram is shown in Figure 1.

Only five jumper connections (ENBL, PCM1 to P1A, PCM2 to P1C, NCM1 to P1B, and NCM2 to P1D) are necessary to configure the H-Bridge to operate in ECCP Full-Bridge mode. An optional sixth connection from the H-Bridge SNSE (current sense) output to the MCP6022 op-amp's input can be attached if you want to monitor the motor's current consumption.

The voltage output at the MCP6022's output is equal to the motor's current consumption. For instance, if the voltage at the output of the op-amp is one volt, the current consumption is one amp. That's in a perfect world as you must consider even with 1% components all around the op-amp, there will be some small percentage of deviation in the voltage measurement.

The fifth jumper connection involves activating the H-Bridge's ENBL (Enable) input pin. The H-Bridge ENBL pin must be logically high to allow the MOSFETs to be driven. The ENBL line can be simply tied high or tied to a PIC16HV616 pin for firmware control. For the sake of simplicity, I chose to tie the ENBL line high with a jumper from the ENBL pin to the +5V pin on the H-Bridge interface.

Once the PCMx and NCMx jumpers are in place, the H-Bridge driver code is used to invoke the ECCP Full-Bridge mode. You may wish to consult Figure 1 again as I describe the ECCP Full-Bridge Forward and Reverse modes. In ECCP Full-Bridge Forward mode, the PIC16HV616's P1A pin is driven active and P1D is modulated via PWM. The active state of the PIC16HV616's ECCP I/O pins is determined in firmware.

In this application, the active state programmed as a logical high. The PIC16HV616's P1B and P1C pins are held inactive in ECCP Full-Bridge Forward mode. To drive the motor in the reverse direction, the ECCP Full-Bridge Reverse mode is used. In ECCP Full-Bridge Reverse mode, the PIC16HV616's P1C

Listing 1

```
void main()
{
        TRISC = 0b11000011;      //P1A-P1D outputs
        ANSEL = 0b00000100;      //select AN2
        ADCON1 = 0b01010000;     //ADC conversion clock Fosc/16
        PR2 = 0x3F;              //set PWM period 320uS@8MHz or PWM freq of 31.25KHz
        OPTION = 0b00000111;     //set TMR0 prescaler 1:256

        ADCON0 = 0b00001001;     //enable ADC
        CM1CON0 = 0x07;          //disable comparators

        TMR2ON = 1;              //turn on PWM
        master_timer = 0;        //initialize clock
        secs = 0;
        mins = 0;
        hours = 0;
        T0IE = 1;                //enable TMR0 interrupt
        PEIE = 1;                //enable peripheral interrupts
        GIE = 1;                 //enable global interrupt

        PORTC = 0x00;

        while(1)
        {
                timer = 0;
                GODONE = 1;
                motor(rev);
                while(timer == 0);

                timer = 0;
                GODONE = 1;
                motor(fwd);
                while(timer == 0);

        }
}
```

LISTING 1. What you don't see here are the motor function and the clock interrupt service routine. Don't worry. The complete volume of source code is included with the H-Bridge driver download package.

pin is driven active and the PWM modulation is provided by the P1B pin. ECCP pins P1A and P1D are held inactive in ECCP Full-Bridge Reverse mode.

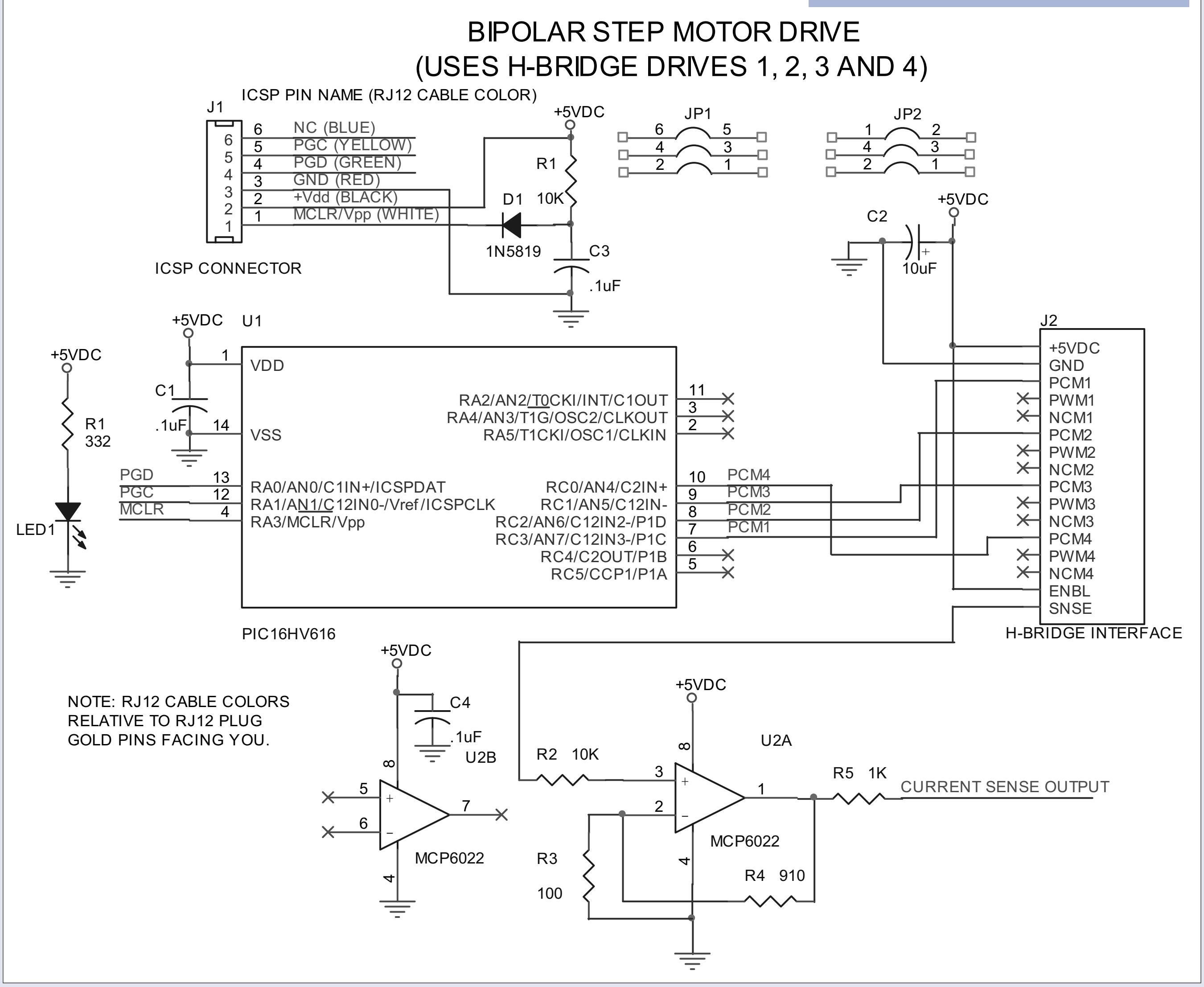
The code needed to drive a simple brushed motor is rather simple and is shown partially in Listing 1. The brushed motor driver firmware configures the PIC16HV616's P1A, P1B, P1C, and P1D pins as outputs, which is required if ECCP mode is to be used. The 10K pot — whose wiper is attached directly to the PIC16HV616's AN2 analog-to-digital converter input pin — will be used in this application to control the speed of the motor. The PIC16HV616's analog-to-digital converter reads the voltage at the 10K pot's wiper and the measured voltage value is then loaded in as the PWM duty cycle value via the CCP1CON and CCPR1L registers.

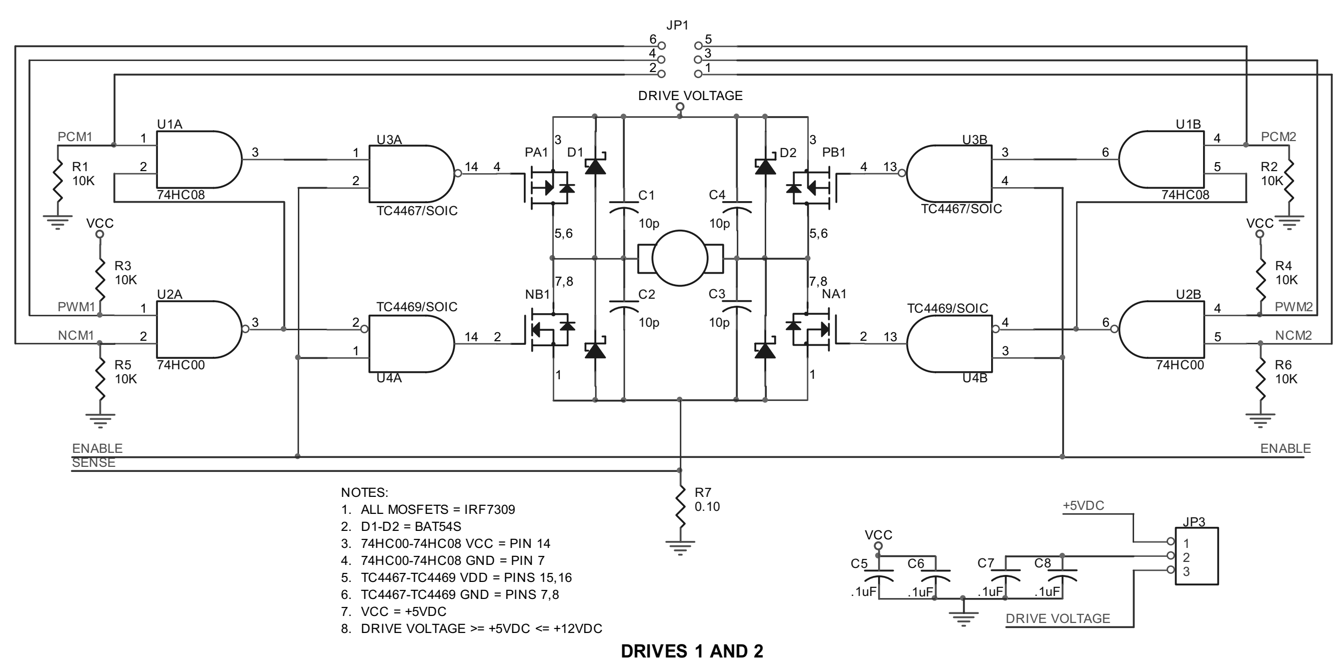
The forward and reverse motion of the brushed motor is determined by bits within the CCP1CON register. The PIC16HV616's internal clock is set for 8 MHz operation. I've also implemented a 32.768 ms-per-tick clock by prescaling Timer0 with a 1:256 ratio. The Timer0 clock is used strictly for delay timing in this application. The brushed motor direction delay is set for 30 32.768 ms ticks, which is approximately one second.

Thus, here's how the brushed motor firmware flows:

- The 32.768 ms tick timer is reset to zero.
- A duty cycle analog-to-digital converter conversion is started.
- The motor function is called with a fwd argument.
- Duty cycle and direction information is loaded.
- P1A is driven active and P1D emits PWM at measured duty cycle.
- Motor shaft turns in a forward direction for one second.
- The 32.768 ms tick timer is reset

SCHEMATIC 2. This is a single-step stepper motor implementation. Note that there is no PWM involved in the stepping motion. PCM1 through PMC4 simply walk through the bipolar step pattern continually.





SCHEMATIC 3. Here's a detailed look at a pair of H-Bridge drivers. Placing jumpers on JP1 makes things interesting.

to zero.

• A duty cycle analog-to-digital converter conversion is started.

• The motor function is called with a rev argument.

• Duty cycle and direction information is loaded.

• P1C is driven active and P1B emits PWM at measured duty cycle.

• Motor shaft turns in a reverse direction for one second.

• This entire cycle repeats from the beginning.

Listing 2

```
void main(void)
{
        TRISC = 0b00000000;           // PORTC = Output
        OPTION = 0b00000111;          // TMR0 Prescaled 1:256
        master_timer = 0;             // Clear RTC regs
        secs = 0;
        mins = 0;
        hours = 0;
        T0IE = 1;                     // Enable TMR0 Interrupt
        PEIE = 1;                     // Enable Peripheral Interrupts
        GIE = 1;                      // Enable Global Interrupt

        while(1)                      // Loop Forever
        {
                timer = 0;            // Clear timer count
                PORTC = PCM1;         // Energize winding
                while(timer == 0);    // Wait for timer to expire
                timer = 0;
                PORTC = PCM4;
                while(timer == 0);
                timer = 0;
                PORTC = PCM2;
                while(timer == 0);
                timer = 0;
                PORTC = PCM3;
                while(timer == 0);
        }
}
```

That's all there is to driving a brushed DC motor. Now, let's turn our configuration and coding efforts towards driving a simple bipolar stepper motor.

Driving a Stepper Motor

Before we can attach a bipolar stepper motor to the H-Bridge Drives, we must do some physical reconfiguring. Note that in Schematic 2, JP1 and JP2 are populated with jumpers. The presence of these jumpers combines the quad of half-bridges into a pair of full-bridges. Let's use Schematic 3 to step through what happens when the JP1 jumpers are in place. Keep in mind as we're walking

LISTING 2. The H-Bridge circuitry takes the complexity out of this code. All we have to do to move the stepper motor shaft is ollow the coil activation/deactivation pattern for a bipolar stepper motor. Operation of bipolar stepper motors is described very well in Microchip's AN907 app note.

through this process that the half-bridges coupled by JP2 behave exactly the same way.

Let's begin by placing an imaginary jumper across JP1 pins 1 and 2 only. When a logical high is presented to the PCM1 pin, the imaginary jumper we just installed on Schematic 3 routes the high-going signal applied to PCM1 over to NCM2, which just happens to be the PA1's electrically complementary MOSFET. NA1 activates and current flows through PA1, the motor winding, and NA1. Now let's add another imaginary jumper across JP1 pins 3 and 4.

When a logical high PWM signal is applied to PWM1, that same signal is applied to PWM2 by way of our newly installed imaginary jumper at JP1 pins 3 and 4. The logical high PWM signal we just applied has no effect upon the output of NAND gate U2A as its input levels did not change. PA1 is still active at this point. At NAND gate U2B, there is also no change in output as PWM2's and NCM2's inputs did not change state either.

When the PWM signal swings to a low logical level at PWM1, it also swings low at PWM2. U2A's output level does not change as both of its inputs are low, resulting in a high output, which keeps PA1 energized. The resulting low-going PWM pulse results in a pair of low-level inputs at NAND gate U2B. U2B's output shifts from logically low to logically high. U4B's inverting input sees the U2B high output as a low input and turns off NA1. PCM2 and NCM1 are kept out of the picture by their pulled down inputs.

When we install that last imaginary jumper across JP1 pins 5 and 6, PB1, NB1, PCM2, PWM2, and NCM1 are drawn into the mix and respond to logic level stimulus exactly like their counterparts PA1, NA1, PCM1, PWM1, and NCM2.

Okay, while we're in imaginary mode, replace the motor in Schematic 3 with a coil from a bipolar stepper motor. Then create another circuit just like the one in Schematic 3 for the second coil of a bipolar stepper motor. What you end up with is an H-Bridge for each bipolar stepper motor coil, or our H-Bridge hardware.

The code in Listing 2 energizes the bipolar motor coils in a pattern that flips the rotor of your stepper motor round and round. To reverse the direction of rotation, simply reverse the order of the writes to PORTC in Listing 2. Speed up the rotation by shortening the delays between steps. Conversely, slow down the rotation by increasing the delays between steps. I used the same 32.768 mS tick timer code from the brushed motor application in the stepper code to make delay generation easy.

If you need to know more about how bipolar stepper motors work, I suggest getting a copy of Microchip's AN907, *Stepping Motor Fundamentals*. There you will find a bipolar stepper motor truth table that you can directly correlate to the stepper motor code and schematics I've provided for you.

Things to Play With

Our H-Bridge and motor discussion is complete. You now have everything hardware and firmware you need to spin small- to medium-sized stepper and brushed DC motors. Here are some things you can tinker with once you get your motors spinning. If you need over-current shutdown capability, you can use the current sense circuitry to feed an analog-to-digital converter input or comparator input on the PIC16HV616 and kill the Enable line when a preset current level is exceeded. Motor RPM can easily be obtained and controlled by monitoring the motor shaft rotation optically and counting the incoming pulses over a predefined time period.

The H-Bridge design used in this series was designed with a development board mentality. You can greatly decrease the physical footprint of the H-Bridge and H-Bridge driver circuits described within these pages to fit a medium-powered programmable motor controller into the tightest of spaces. SV

SOURCE

PIC16HV616
MCP6022
Microchip Technologies
www.microchip.com

Peter Best can be contacted via email at peterbest@cfl.rr.com

For years, I always wanted to attend the Robot Fest in Baltimore, MD, but something had always interfered with my plans to attend. This year, I decided nothing would stand in my way. Gary Mauler — one of the organizers of the event — set me up with a nice booth with lots of space in a very key location at the event. With Gary going to all that trouble, I couldn't let him down. I had to bring something to knock the socks off all the attendees.

by Michael Simpson

FaceWalker

Part 1 — The Foundation

Several years ago, I attended a modern art show and was impressed with a piece that had incorporated a CRT and a video loop of an animated human face. It was so eerie that I found myself constantly staring at the piece throughout the show.

My idea was to build a walker robot with a human face that could react to the spectators.

Before I get into the construction of the FaceWalker, I want to tell you about the effect it had at the show.

From the time I walked into the show and set the FaceWalker on the table, I had a packed crowd around my booth. The FaceWalker was so popular that I had individuals waiting 20 minutes while I charged batteries just so they could get a second look at the robot in action.

What made FaceWalker so unique was the fact that the main controller was a Pocket PC that had an animated human face with various expressions, as well as sound that was timed to the mouth and eye movements. When I placed the FaceWalker in attack mode, the face would show a rather nasty expression and growl.

FaceWalker Construction

There are three key components that make up the FaceWalker:

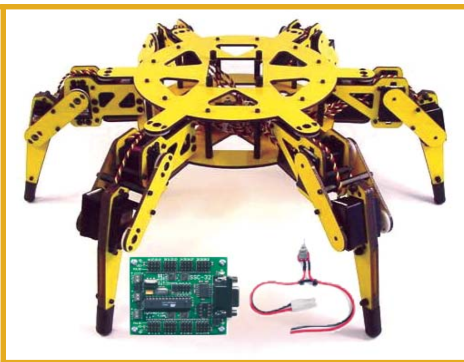


Figure 1

Figure 2

• The base, which consists of the Lynxmotion EH3-R and a SSC-32 servo controller and support components such as batteries, connectors, and the Pocket PC mount.

• A controller and controller interface which consists of a wireless PS2 controller and DiosPro microcontroller.

• A 500 MHz IPAQ Pocket PC which is used for the brain.

I want you to be able to build the FaceWalker or something similar, so I am going to break down each key component in a separate article. This will allow me to provide enough detail for you to recreate that component for your own project. In this article, I will concentrate on the base and its construction. Next month, I will provide you with the instructions to build the controller interface, and the last article will provide you with the details I used to build the brain.

We have a lot to cover and only a few pages to do it in, so let's dig in and get our hands dirty.

Base

No matter what kind of robot you build, you have to build a base that is strong enough to hold all the components and batteries. I calculated that I would need a base that would support almost 10 lbs of total weight. For a walker, this consideration is more critical than a track- or wheel-based robot since the legs have to support the weight and still be able to articulate.

While I was searching the Web, I found plenty of walker bases, but none stood out like the Lynxmotion EH3-R Hexapod base shown in Figure 1. The EH3-R is a special round version of their Hexapod. The round base gave me two distinct features that I wanted in this particular robot.

1. I wanted to be able to move the robot in any direction independent to the direction it was facing.

2. I wanted the robot to look like a spider.

Originally, I thought I could use standard servos to control the robot, but after talking to Jim Frye at Lynxmotion, he assured me that standard servos just did not have the power to handle the kinds of force that I was going to place on them.

I decided to go with the EH3R and 18 HS645 servos. The HS645 servos are much more expensive, but I wanted something that would hold up to the payload and speed I was going to put the robot through at the various shows and demos I would attend.

I only had two weeks until the show so, while I waited for the EH3R to arrive, I started sourcing other components I would need for the base. I will list the source and location for all the items I used to create the FaceWalker base at the end of this article.

Figure 2 shows the additional components I needed: 7.2V Stick battery, six-cell AA battery holder, 9V clip, six AA batteries, SPDT switch, and a Velcro strip. Most of these items can be obtained at your local hardware store or nearest RadioShack.

You will also need some additional hardware, which is shown in Figure 3. The hardware is used to attach a platform on which we will install our Pocket PC mount. You will need 22 #4 machine screws, 20 #4 lock washers, and six hex nuts. You will also need some 1-1/2" standoffs, but because they are hard to find, I used 11 #4 M-F .5" and 11 #4 F-F one inch long to create what I needed.

The platform on which we will attach our Pocket PC mount will need to be at least 7" in diameter. You can also create a bit more of a bug shape, as shown in Figure 4.

It's important to note that you can use any material you wish for this platform. However, if you are going to use a Pocket PC mount with a suction

Figure 3

Figure 4

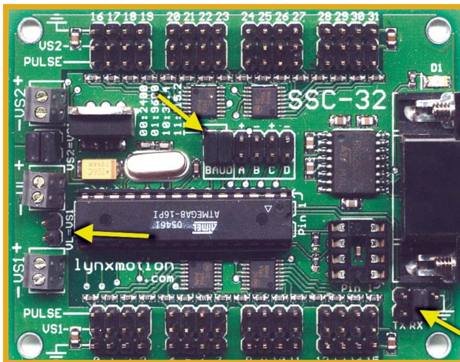


FIGURE 5

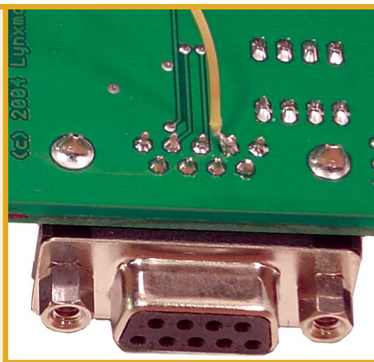


FIGURE 6

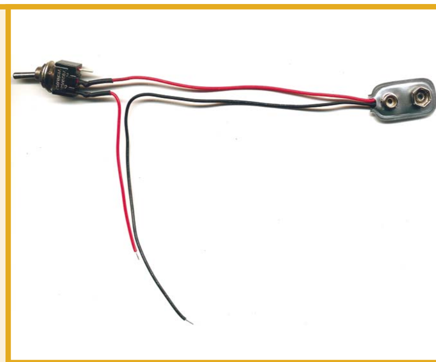


FIGURE 7

cup, you will want as smooth of a surface as possible. I used 1/8" clear Lexan. I then painted the underside black.

SSC-32 Prep

Before we start assembling the base, the SSC-32 needs to be prepped. First, set the jumpers as shown in Figure 5. You will need to remove the VL=CS1 jumper and the TX jumper. Place both the baud rate jumpers to set the speed to 115200.

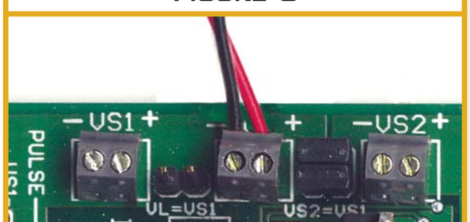
Take about 12" of hookup wire and strip about 1/8" of insulation off one of the ends. Attach this end to the back of the SSC-32, as shown. This is the DTR pin on the nine-pin cable. This will be used to tell the PS2 controller that we want a reading and will be connected to the interface in Part 2 of the FaceWalker.

The FaceWalker has two power sources: an RC Stick battery is used to power the servos and a set of six AA batteries are used to power the logic and various other components, such as the wireless controller. You will want to be able to switch this power source on and off so a switch will have to be wired in series with the 8" 9V battery clip, as shown in Figure 7. Use some shrink to insulate the connections on the switch. This adds reinforcement, as well.

Once you create the logic power harness, connect it to the VL header as shown in Figure 8.

At this point, you can proceed with the EH3R base assembly.

FIGURE 8



Base Assembly

Figures 9 and 10 show the EH3R Lexan and hardware. The 18 HS-645 servos are packaged separately. You will need to download the assembly instructions from the Lynxmotion website at:

- Leg Assembly Instructions — **www.lynxmotion.com/images/html/build38b.htm**

- Body Assembly Instructions — **www.lynxmotion.com/images/html/build42b.htm**

Before doing any assembly I recommend that you do the following:

- Read the information about Polycarbonate on the Lynxmotion website at **www.lynxmotion.com/images/html/infolexa.htm**

- Remove and separate all pieces from the sheets. The individual pieces are laser cut from Lexan sheets. The pieces are still attached to the sheet and need to be separated. You could do this as you assemble, but it will slow you down.

- Remove the protective plastic covering from the Lexan surfaces. If you wait till after assembly, it is much harder to remove.

- Punch out all the small holes. There will be small Lexan pieces in each of these holes left over from the cutting process. Again, it is much easier to do this all at once.

The first thing you will need to do in the assembly process is to put the legs together. Refer to the leg assembly instructions. You will need three right legs and three left legs, as shown in Figure 11. The left legs are a mirror image of the right, but I got a bit confused as I was

FIGURE 9

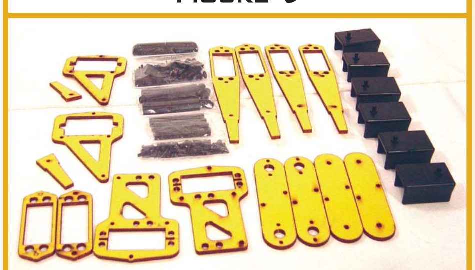


FIGURE 10

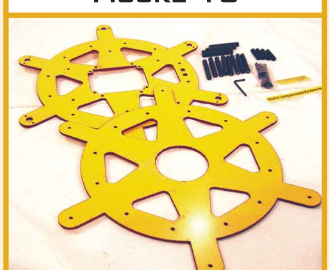


FIGURE 11

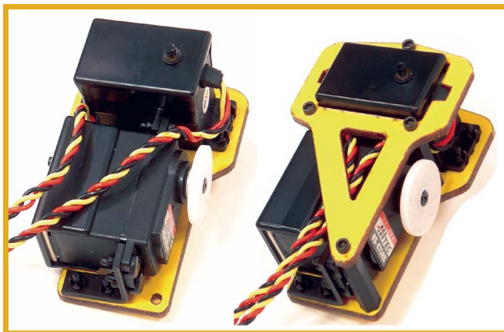


Figure 12

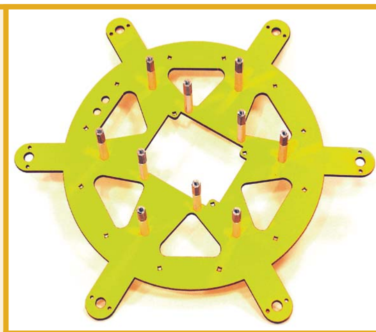


Figure 13

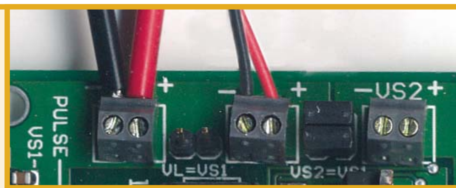


Figure 14

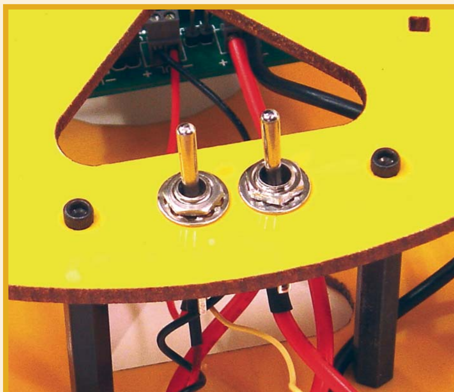


Figure 15

building them.

Figure 12 shows a partial assembly for the left legs. This should help.

With the legs built, we can move on to the body. Again, you will need to refer to the body instructions mentioned previously. Before you start with the body, connect the 1/2" standoff to the 1" standoff to create a 1.5" standoff and attach it to the top side of the top panel as shown in Figure 13.

Six of them are attached to the middle ring of holes. The remaining four connect to the four holes near the center square — not the corner holes — which are used to mount the servo controller. Notice that the top panel has a slight cup. These standoffs connect to the convex side of the cup. Later we will attach our platform to these standoffs.

When you get to Step 3 of the body instructions, do *not* install the VL=VS1 jumper. Just connect the R/C battery harness as shown in Figure 14.

In Step 4, install the Logic Power switch into the first hole as shown in Figure 15.

Once you have completed the construction of the body, you'll need to make a few finishing touches. First, you will need to create a battery compartment for the Stick battery. This is done by cutting a 9" section of hook and loop (Velcro). You will need to attach the hook portion to the loop portion with some hot glue, as shown in Figure 16.

Once the two pieces are connected, add a small loop to the end of the hook side. Note that the loop section is the piece that feels like soft carpet.

The key here is to attach the loop that you made around the first standoff, then weave the Velcro around the other posts as shown in Figure 17. This creates a nice padded battery compartment. You should dry-fit the pieces of hook and loop before you actually hot glue them in order to get the placement correct. You may find it easier to add the small loop first.

Next, you need to create a small compartment to hold the six-cell logic battery. This is done by connecting a piece of Velcro to the posts to the left of the switches, as shown in Figure 18. Wrap a hook piece around one post and secure it with hot glue. Then wrap a loop piece around the other post and secure it. Dry-fit first so that you can get the orientation of the hook and loop correct. Next, route the 9V clip into this space. Now all you need to do is connect the battery clip to the six-cell pack and slip it

Figure 16

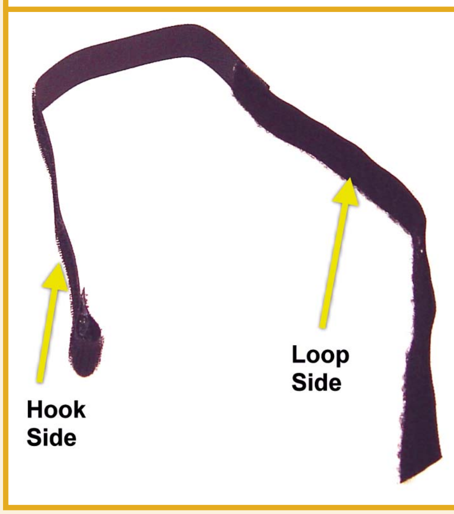


Figure 17

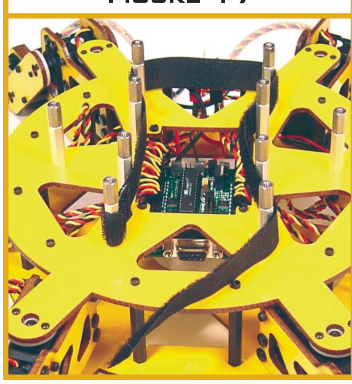


Figure 18

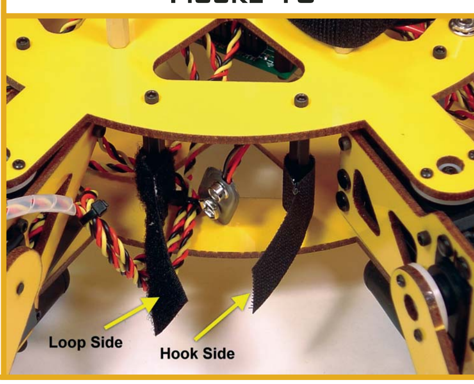
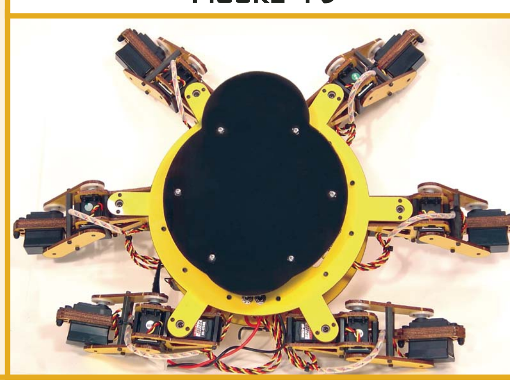


Figure 19



Parts List

Item	Supplier	Part No.
• Round Base with 18 Servos	Lynxmotion	#EH3R-KT
• HS-645 Upgrade	Lynxmotion	#SUP-04
• Servo Controller	Lynxmotion	#SSC-32
• Wiring Harness	Lynxmotion	#WH-01
• 22, #4 3/8" Machine Screws	Jameco	#40969CK
• 20, #4 Lock Washers	Jameco	#4106850CK
• 6, #4 Hex Nuts	Jameco	#40942CK
• 11, #4 M-F .5" Standoffs	Jameco	#111755CK
• 11, #4 F-F 1" Standoffs	Jameco	#139184CK
• SPDT Switch	Jameco	#22832CK
• Six-Cell Battery Holder	Kronos Robotics	#16321
• 9V Battery Clip (8")	Kronos Robotics	#16264
• 5, Heat Shrink Strips	Kronos Robotics	#16287
• 7.2V 2000 mAH – 3000 mAH Stick Battery	Check local hobby shop, RadioShack, or Lynxmotion.	
• 6, 1.2V – 1.5V AA Batteries Rechargeable or Alkaline	Check hardware store.	
• Velcro Strip — 3/4" x 1 yard Should be the non-sticky type.	Check fabric section of local store.	
• 8" x 10" Plexiglas or Compressed PVC	Check local hardware or home center.	
• Universal PDA Mount	Check auto store or **Amazon.com**	
• Hookup Wire 22-26 Ga (Different colors will be helpful.)		

NOTE: Do *not* use a battery that supplies more than 7.2V or you will burn up the servos! Even at 7.2V we are pushing the servos.

- KRMicros — **www.krmicros.com**
- ZeusPro — **www.krmicros.com/Development/ZeusPro/ZeusPro.htm**
- Lynxmotion — **www.lynxmotion.com**
- Jameco — **www.jameco.com**
- Kronos Robotics — **www.kronosrobotics.com**

into the small space. Connect the two pieces of Velcro to hold the pack in place.

The last step is to connect the platform to the completed body. The platform needs to be 7" in diameter, as a minimum. You can make the shape whatever you wish. I created a segmented shape for mine, shown in Figure 19. If you are using clear Lexan, the attachment is very simple. Place the platform on top of the standoffs and mark the six outermost posts. Don't create holes for the inner four standoffs as they may get in the way of your Pocket PC mount. Once marked, drill 1/8" holes and attach with #4 machine screws. Do *not* paint the platform yet as we will be tooling it a bit more in Parts 2 and 3.

While we don't have our Pocket PC or PS2 controller connected, there are a few things we can do to test our FaceWalker. I have included two test programs — which are available on the *SERVO* website (**www.servomagazine.com**) — that will allow you to connect your desktop or laptop to the walker with a serial cable. The first program is called SSC32Test. This program lets you set any servo connected to the SSC32 to any position. The second program is called FaceWalker1. This program is a desktop program that will allow you to put your walker into motion. The speed of the walker has been fixed at a slow pace as the program is meant for testing only.

What's Next

In Part 2, I will add an interface that will allow you to plug a PS2 controller into the FaceWalker. In Part 3, we will finish up construction and I will show you how to add a Pocket PC to the base. I will go into detail about the software so you can add your own special actions.

The Zeus source code used to create the two programs, as well as project updates, can be downloaded from the Kronos Robotics website at **www.kronosrobotics.com/Projects/FaceWalker.shtml**

ZeusPro is a very simple and inexpensive way to create both Windows and Pocket PC software. Please be aware I won't be going into any details about the software until Part 3 of the series. **SV**

When a Hammer Won't Work

Yes, believe it or not, a hammer is not the perfect tool for every occasion. Although you'd probably want to take a hammer to some of the fasteners finding their way into more and more consumer electronics products. Take the Game Boy Advance (GBA), for example.

Your biggest obstacle won't be building high scores on the video games for the GBA. Your hardest challenge will be getting inside the GBA case. Any hacker worth his or her weight in solder trying to open up the GBA will be stopped dead in their tracks.

In an attempt to clearly tick-off hackers, Nintendo assembled the GBA case with six Triwing® "tamperproof" screws. Now don't be fooled into thinking that you can simply push a straight-blade screwdriver into these screws and force 'em out. What you really need for opening the GBA case is a Triwing screwdriver.

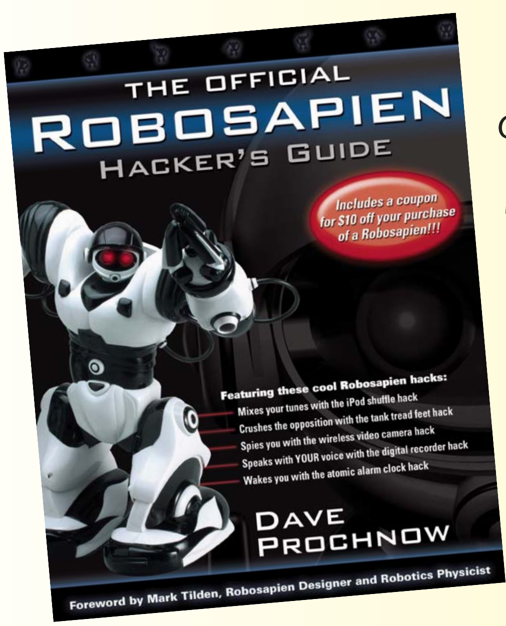
Sears, Roebuck, and Co. sells the Craftsman 32 pc. Security Screwdriver and Bit Set (Sears item #00947486000 Mfr. model #47486) for less than $25 which features four Triwing bits. Additionally, you will find nine Torx® bits (T7 through T40), 11 fraction and metric hex bits, four spanner bits, and three Torqset® bits. This is the ideal assortment for opening up everything from notebook computers to kids toys.

And remember this old hacker trick. Once you've expertly removed a tamperproof security fastener, make sure that you replace that cantankerous bit with a more "regular" Phillips screw. Then you can save that hammer for "persuading" a stubborn axle out of sticky wheel hub.

Sears makes a set of tamperproof bits that can easily remove most of your security fastener headaches.

Only 147 Days Until This Present is Gone

If you're looking for some more cash savings on the recently reduced price of the WowWee Robotics Robosapien™, then look no further than *The Official Robosapien Hacker's Guide* (McGraw-Hill, 2006). Inside the back cover of this book, you will find a $10 rebate coupon that is applicable towards the purchase of any WowWee Robotics product. This includes the remarkable Robosapien which is currently priced less than $60. Do the math; that means that you can purchase one of the most powerful robot brains on today's market for less than $50. You can purchase this book from the *SERVO Magazine* Online Store. (Did I mention that I wrote this book?)



Get inside this lug's head and do it by the book.

Reason in the Sun

With summer in full swing, you'd better make sure that you have ample batteries for powering all of your outdoor bot activities. Rather than filling the andfills with dead dry cell batteries, you might want to consider rechargeable batteries. And there's no better way to charge batteries than for free. Right? Well, as free as the sun, away.

The particular model of solar battery charger that I regularly use can accept two D size batteries (or, C, AA, and AAA) at a time. Likewise, the intensity of sunlight can dramatically affect the battery charge time. So strive for maximum sunlight and give yourself about 9-18 hours for each pair of batteries. A nifty meter is built into the solar battery charger sold by C. Crane. This meter will help you estimate your charging time.

For example, a solar intensity of 120 mA will take approximately 12 hours worth of charging. In order

A solar-powered battery charger is a great tool for making your bots both more fun and lower-cost to operate.

to obtain the maximum charging capability, you can use the built-in prop stand on the bottom of the charger. This stand will help you achieve the best "angle of the dangle" for keeping your charger's best face towards the sun.

The most common fault with these solar battery chargers is inadequate sunlight intensity. For example, robot experimenters in the north (e.g., North Dakota) will need a greater length of time to charge their batteries. Don't be alarmed if your charger's meter indicates a solar power current of 80 mA or less. Bot builders along the Gulf Coast can safely down a Mint Julep (or three) waiting for four batteries to charge. In this case, I have easily obtained solar power current readings of 160 mA.

If this lengthy power charging time commitment really bugs you, consider buying two solar battery chargers. Oh, and don't forget the sunscreen.

You Say It's Your Birthday, We're So Happy fer Ya

Cue the drum roll, the LEGO® Mindstorms® NXT robot design kit has now been officially released to its adoring public. In one of the best kept secrets in the robot world, a forthcoming book about the NXT system is being marketed with a special LEGO brick kit.

This brick kit was assembled with the newest version of LEGO Digital Designer (LDD). By using LDD, a kit was designed that will be sold through the LEGO Factory online gallery.

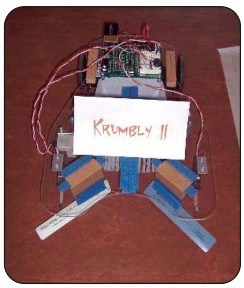
So what does this kit include? There are over 300 pieces that will enable you to build some of the special projects that are discussed in the book. Psst, can you say R. Buckminster Fuller, Rem Koolhaas, and LOT/EK? No? Well, you'd better rush to your favorite online search engine and get a sneak preview. Keep an eye out for this book in the *SERVO Magazine* Online Store. It should be arriving just in time for Christmas. (Did I mention that I wrote this book, too?) **SV**

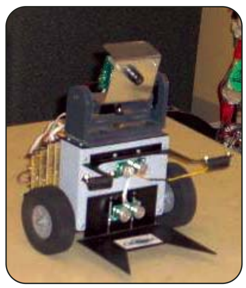
Menagerie

The 4th Annual HBRC Challenge

by Camp Peavy

It was beautiful weather for the 4th Annual HomeBrew Robotics Club Challenge; important because the first event was RoboMagellan. This year's Phase I had two RoboMagellan entries: Rusty and Xploradora. Rusty did well as he eventually made it to the target GPS coordinate; however, he needed a little help with the patio and hose traps. There were actual water-sprinklers going off in the bonus cone area. One of these days it will actually be raining; remember, it doesn't have to be waterproof ... just water-resistant. The color camera (CMUcam-1) wasn't yet integrated (Phase II), so Rusty got as far as his program allowed. Brandon Blodget's Xploradora's claim-to-fame is the first robot to actually touch the orange cone in an official competition (Portland 2005). However, on this night, Xplora's computer wouldn't boot. Anyone who builds robots has been there ... so Brandon gave us a nice overview of the system and some insight into his strategy, plus a rollicking account of his RoboGame experience where Xplora took Bronze as part of an HBRC sweep. Bob and Ted (Odyssey) from Ologic took Gold; Jim and Doug (A3) took Silver, and Xploradora took Bronze; homebrewers all.





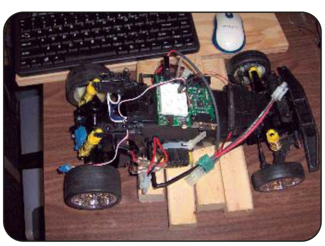
Next event was the TABLEBot Challenge; the not-a-contest that started it all. Phase I is simply driving your robot from one end of the table to the other and back. First, I ran Timmay as a demonstration. After a few cheesy re-arrange moves (remember this is not a contest ... it is a challenge and we want to show your robot in its best light), Timmay eventually put the block into the shoe-box.

Next was Henry Meier with Tribot. The little BoE-based robot dutifully went from one end of the table to the other and back again. Phase I success for the Tribot and Henry.

*The HomeBrew Robotics Club **www.hbrobotics.org** meets the last Wednesday of every month (except December) at Carnegie Mellon University-West located in Mtn. View at the NASA Research Park of Moffett Field, California.*

Tony Pratkanis was the next participant; Tony has participated in every HBRC Challenge since its origin in 2003. This year's entry is yet another unpronounceable "ant" analogue Myrmecia Pilosula or the jumper ant (TABLEBot ... jumper ... get it?). Vex-based Myrme made it from one end of the table to the other and back and thereby completed Phase I ... without jumping! Next, Mike Thompson demonstrated his hex-legged creation JIT (Just in Time). JIT meandered up one side of the table to the other and back again; amusing everyone ... but then again we're easily amused.

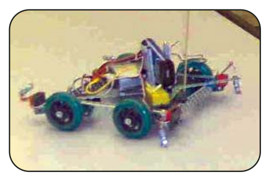
Next Andrew and Matthew Downing demonstrated DHOFT (Didn't Have Enough Time or something to that effect). DHOFT is a Vex-based robot with an ingenious and simple ledge sensor. What Andrew and Matthew did was simply dangle a weight from a couple of limit switches and when those weights dropped, the switches triggered, signaling to the 'bot that beyond thar be dragons! DHOFT successfully went from one end of the table to the other and back successfully completing Phase I, as well.



Next Dr. Robot, a.k.a., Alan Federman showed a reflexive sensor that he built for his radio-controlled Vex-based robot named DropTest. He drove the robot end-to-end and each time the robot encountered the edge, it automatically stopped and moved back an inch. Nothing in the rules say the robot can't be remote controlled. Next up, Mike and Rose once again demonstrated what has become one of our favorite homebrewed characters — Ingoshu. Ingoshu dutifully made it from one end of the table to the other and back. In fact, Mike found a short-cut as the rules don't specify length or width. Hey! It's all just fun!

Next up, Al Margolis of Hobby Engineering demonstrated a robot named Procrastination. Procrastination performed Phase I right on schedule. Ted Larson of Ologic demonstrated a perennial club favorite named Tracker. Tracker successfully navigated the table for Phase I, pushed the block off the table for Phase II, and with a little coaxing, managed to drop the block into the box for Phase III. All in one night! Conner Brew was next with Krumbly II. Krumbly took *gold* at the recent RoboGames for Best of Show; Junior League. Not only was Krumbly II able to sweep the table clean, but was able to push the block off the table; successfully completing Phase I. Good job, Conner!

Continuing with the excellent adventure, Bob Allen with Ologic went next with his TABLEBot named Tormax. Tormax successfully completed Phase I. Along the way, Bob demonstrated Tormax's dead-reckoning capability by holding one side back and then the other; each time the robot would correct the difference and continue on in a straight line. This robot's got PID. Next club president Wayne Gramlich brought out a custom-milled RoboBrick-based TABLEBot named The Edge of Darkness utilizing a homebrewed CAN-based bus Wayne developed over the past year from a discussion on the HBRC mailing list. After the TABLEBot category, it was time for the Open Exhibition portion of the Challenge. I started off with a brief demo of my three robots Springy Thingy, The PROTOBot Army, and Wirey. Springy autonomously followed the IR beacon and shook hands (her two tricks); the PROTOBot Army has a new addition since the RoboGames ... a ribbon climber, and Wirey has a new response; whenever it encounters an obstacle, be it human or wall, it says, "Hello, I am your friend ... let's play!"



Next Ben Margolis with Hobby Engineering demonstrated the functioning Propeller-based robot, Backup. Ben says they're going to turn it into a RoboMagellan robot. Excellent! Next Dennis Burke showed a sweet little LCD screen that he's playing around with.

Lastly, Tony Pratkanis showed his Gold-Medal winning Trinity Firefighting robot Solenopsis invicta (fire ant).

That's it for our 4th Annual HBRC Challenge. I want to thank all of the participants. Keep those soldering irons hot!

by Dave Calkins

ROBOGAMES 2006

International Event Draws Thousands

They came from Russia. They came from Singapore and Japan. They came from Brazil and Columbia. From England and Switzerland. From across Canada and the US. Engineers from 20 countries came to RoboGames. They proved that engineers *can* be cool and even manage to put on a great show.

There were 42 total events and over 400 robots. Robots walked, rolled, spit fire, pneumatically actuated, slithered, solved puzzles, strove for peace, fought epic battles, and some were even powered by steam.

As is typical at RoboGames, the big draw for the public was the Combat events. Nine classes ranging from 5 ounces up to 340 pounds fought to the thrill of the crowd. Nearly half the bots that came were combat robots. (Does that mean we're fighters, not lovers?) While the US had the most entries, four Brazilian teams showed up with six robots and an unlimited supply of enthusiasm. Robot Combat is now a major force in Brazil, and they brought the best of the best to the US.

"It doesn't matter if we win, we're here to have fun!" said Brazilian group leader Paulo Lenz. The Brazilians had more than just fun however, managing to take home a gold medal and two bronzes.

The best battles of the weekend were put on by the recurring fights between Sewer Snake (USA) and Ziggy (Canada). Youtube is now filled with videos of the two combatants — Ziggy, a 340 pound flipper, continuously chucked the 220 pound Sewer Snake eight feet up and 20 feet across the arena like it was a soccer ball. The 'bot came down hard every time, to the screaming cheers of the packed stands.

Though #1 ranked Sewer Snake should have been fighting other Heavyweight robots, driver Matt Maxham was looking for a challenge, and so he and wife Wendy registered all their robots a full class above their nominal weight. They didn't do well, but in the spirit of the event, they had a ton of fun. Well, 220 pounds of fun, anyway.

The combat arena sat right next to the Tetsujin platform, a *SERVO*-sponsored event first held in 2004 at RoboNexus. Monty Reed and Alex Sulkowski gave demonstrations with the exosuits they brought. We've all long imagined about exoskeletons that could lift cars (or maybe superheavy-weight robots), and these guys are making Robert Heinlein's dreams come true.

Along with weight-lifting demos, Monty also did some walking. That may not seem like much, but it was the suit that provided the force to walk, not Monty. It was a great moment when Monty started hopping across the combat arena floor using only the power of his pneumatic suit! This is the type of technology development that can lead to suits that a paraplegic could wear to walk across the room.

Another exciting addition to RoboGames this year was human controlled soccer! We've all seen humanoid Robo-Ones, but do you know what happens when you hack a Hitec RoboNova to run off a Sony Playstation controller and a Xbee WiFi controller?

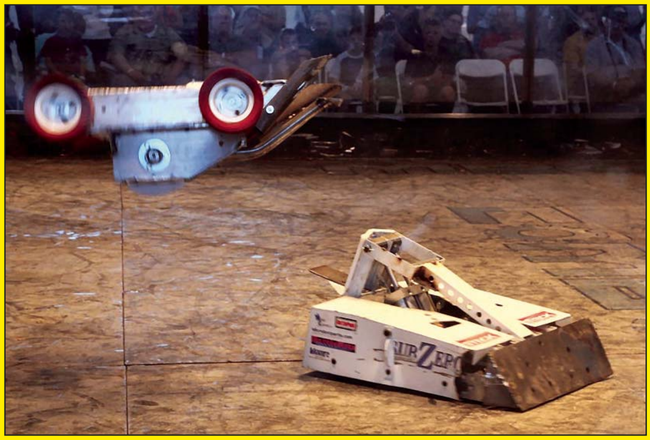
Easier movement, that's what! What better way to show this off than with several games of soccer! The final between the US and the UK had the bleachers filled and a standing-room-only crowd surrounding the soccer pitch.

Like human soccer, the match scores are small, but the atmosphere is tense. Several times the US got to the UK goal, only to miss critical shots or see their shots deflected by the UK goalie. The ball was held at the UK side, seemingly forever, as defenders and strikers alike fought for control of the ball.

Finally, about 22 minutes into the 30-minute game, the US managed a goal on a sidekick, as the striker (at a right angle to the goal) did a crotch-smashing set of splits, kicking the ball around the goalie and into the nets. The crowd exploded in applause, everyone standing up and cheering, proving once and for all that all robot sports are exciting. (The RoboNova robots and modified controllers are both available at **robogames.net**.) Brazil won over Columbia for the consolation bronze medal.

Not all robots need to move to be cool, however, as the Art bots drew a heavy turnout in both entrants as well as entranced audience members. I Wei,

“As is typical at RoboGames, the big draw for the public was the Combat events. Nine classes ranging from 5 ounces up to 340 pounds fought to the thrill of the crowd.”



ROBOGAMES 2006 CHAMPIONS

ROBOT SOCCER — 3:3 Android
1st — USA, ComBots; 2nd — UK, Got Robots; 3rd — Brazil, SohToskeira

Combat — 340 lbs
1st — Canada, Ziggy; 2nd — USA, Sewer Snake

Combat — 220 lbs
1st — USA, Original Sin; 2nd — USA, Brutality; 3rd — USA, SJ

Combat — 120 lbs
1st — USA, Stewie; 2nd — USA, Ice Cube; 3rd — Brazil, Touro

Combat — 60 lbs
1st — USA, Son of Whacky Compass; 2nd — USA, Death By Monkeys; 3rd — USA, Hexy Jr.

Combat — 30 lbs
1st — USA, Killabyte; 2nd — USA, Gnome Portal; 3rd — USA, Bot 6:00

Combat — 12 lbs
1st — USA, Darkblade; 2nd — USA, Bullet; 3rd — USA, Lil Shocker

Combat — 3 lbs
1st — Brazil, Mini-Touro; 2nd — USA, Itsa; 3rd — USA, Titanium Chipmunk

Combat — 1 lb
1st — USA, MC Pee Pants; 2nd — USA, Switchblade; 3rd — USA, Team DMV

Combat — 5.3 oz
1st — USA, Microdrive; 2nd — USA, Change of Heart; 3rd — USA, VD

Sumo — 3kg (Auto)
1st — Singapore, Zero Plus; 2nd — Singapore, Chain; 3rd — Singapore, I Spy

Sumo — 3kg (R/C)
1st — Singapore, Judge; 2nd — Singapore, Zero; 3rd — Singapore, I Spy

Sumo — 1kg (Lego)
1st — USA, Your Luck; 2nd — USA, Reactor; 3rd — USA,Torque

Sumo — 500g (Auto)
1st — Singapore, Sensor Knight; 2nd — Singapore, Grace; 3rd — Singapore, Extreme

Sumo — 100g (Auto)
1st — Singapore, Micro; 2nd — Colombia, UPB Goliath; 3rd — USA, Newton

Open — Best of Show
1st — Taiwan, Steam Tank; 2nd — UK, Ziggy; 3rd — USA, PROTOBot Army

Open — Line Slalom
1st — USA, Twisted; 2nd — USA, Linex; 3rd — USA, Chip

Open — Ribbon Climber
1st — USA, Reach 4 the Sky; 2nd — UK, SER; 3rd — Canada, Mars Lift

Open — Lego Open
1st — USA, Stick; 2nd — USA, Mousetrap; 3rd — Russia, Offroad

Open — Biped Race
1st — Japan, Pirkus; 2nd — USA, RoBozoid; 3rd — Colombia, Osiris-1

Open — Walker Challenge
1st — UK, Ziggy; 2nd — UK, Flik; 3rd — Switzerland, Galileo

Open — Robomagellan
1st — USA, Odyssey Goal!; 2nd — USA, A3; 3rd — USA, Xploradora

Open — Fire-Fighting
1st — USA, Solenopsis Invicta; 2nd — USA, Flame-Out; 3rd — USA, Isis

Open — Table Top Nav
1st — USA, Buggy; 2nd — USA, Next Top

Open — LEGO Challenge
1st — USA, Run Around; 2nd — USA, Triple Step Rocker

BEAM — Speeder
1st — Peru, Speed-O-Bot; 2nd — USA, Perihilion

BEAM — Photovore
1st — Peru, Shadowfobic; 2nd — UK, The Turtle

Robo-One — Wrestling
1st — USA, Kugai; 2nd — USA, Rooks Pawn; 3rd — USA, RN-1E

Robo-One — Demonstration
1st — Japan, Plen; 2nd — USA, Felix; 3rd — USA, Gold Finger

Tetsujin — Weightlifting
1st — USA, Xela4; 2nd — USA, Levo Robot Suit 17

Tetsujin — Walking Race
1st — USA, Access Suit 13b; 2nd — USA, Xela2

Art Bots — Static
1st — USA, Deadly Necrobot; 2nd — USA, Sentry #2; 3rd — USA, Emergence

Art Bots — Kinetic
1st — Taiwan, Steam Walker; 2nd — USA, Slither; 3rd — USA, Pendulum

Art Bots — Musical
1st — USA, Arca Musarithmica 1; 2nd — USA, Singing RGB's

Jr. League (<18 yr old) — 500g Sumo
1st — USA, Push; 2nd — USA, RoboRaptor; 3rd — USA, R2D2

Jr. League (<18 yr old) — 120 lb Combat
1st — USA, Dolor; 2nd — USA, Renegade; 3rd — USA, Swamp Thing

Jr. League (<18 yr old) — Woots & Snarks
1st — USA, Lego Rover; 2nd — USA, Wombat; 3rd — USA, Can-do

Jr. League (<18 yr old) — LEGO Challenge
1st — USA, Run Around; 2nd — USA, Triple Step Rocker

Jr. League (<18 yr old) — LEGO Open
1st — USA, AquaKeeper; 2nd — USA, blitzbot

Jr. League (<18 yr old) — Best of Show
1st — USA, Krumbly II; 2nd — USA, Aquakeeper

was the most popular with his three steam-powered robots. Tired of batteries not holding a charge? Don't want to use an internal combustion engine? Go steam! His amazing steam-punk robots walked and rolled around the venue, delighting adults, children, and the judges (winning gold medals in both Kinetic Art and Best of Show). These robots were unique in every way, and I hope that they inspire more people to consider thinking outside the [battery] box when building.

Static robots included the stunning "Necrobot," which used hydraulic valves, motorcycle parts, and cow bones to create a five-foot tall sinister robotic tyrannosaurus. Al Honig

Sumo (left) and RoboMagellan (right) were just two of the 42 events featured at RoboGames.

created several bots for the event, mostly made from chrome oddities. While none of these bots moved, they did show tremendous strides in robot aesthetics. After all, the goal is to get robot builders to build more friendly looking robots that will be easier to interact with, right?

The fire-fighting competition was once again dominated by 13-year-old Tony Pratkanis, smoking several engineers who were three times his age. While the dynamic duo of Bob Allen and Ted Larson of Ologic did their valiant best, they still lost to a middle-schooler. I'm sure that their egos will survive another 20 years until Tony is only half their age ...

Not all robot competitions are held indoors under fluorescent lighting — RoboMagellan (invented by the Seattle Robotics Society) was also a very popular event, held outdoors in the four-acre park above Fort Mason, with the Golden Gate Bridge and Alcatraz as background.

Eleven robots autonomously roamed the park over hills and down gullies, looking for orange cones and a final GPS point. Larson and Allen found redemption, as their Magellan bot "Odyssey Goal!" finished with the highest score for the gold medal.

The sumo event — the founding competition of RoboGames — was once again dominated by the Singapore team. They swept both 3 kg and the 500 g events, but only managed to win a single medal in the 100 g class. (Only 10 of 12 medals ... what's a team to do?) The new LEGO class was a big hit, letting both kids and adults compete using Mindstorms robots.

The ribbon climber drew teams from around the world, with Colleen Timmins of Canada grabbing the bronze and then grabbing a taxi to the airport a few minutes later! Camp Peavy's ProtoBot army proved their worthiness as the Best of Show bronze medallist when, in only a few minutes, Camp reconfigured some of his ProtoBots to compete in the ribbon climber event. He didn't win, but he did prove that configurable bots are the way to go!

Next year's event will return to Fort Mason on June 15-17th, with even more events. The 2007 FIRA robot soccer championship will co-locate with RoboGames — teams from 30+ countries are expected to compete in 10 different styles of soccer (see the March '06 *SERVO* for details on FIRA). I actually just returned from the 2006 championship in Germany, and I was saddened to say there were no US teams. Therefore, I'm throwing down the gauntlet for *SERVO* readers: Let's see a US soccer team!

And, while I'm on a soapbox, how about more Tetsujin competitors? This is a great event, and needs more participants to continue.

We're also creating a new event: RoboRally. Much like RoboMagellan, R/C-based cars will have to autonomously navigate a course. Unlike Magellan, it won't be GPS based or held on grass. Cars will have to autonomously follow a figure 8 pattern on a scale four-lane street with yellow and white lane-divider lines. The goal will be similar to line-followers, only you'll have to follow lines left to right rather than centered.

Many events at this year's RoboGames had less than five entrants — this is a shame, and I want to see each of you step up to the challenge. Next year's event is offering over 70 competitions! Seventy! Surely you can find the time to build a robot to compete in one of them. By the time you read this, you'll have 10 months to register and 11 months to build a robot. What are you waiting for? Those medals sure do look good, and you get to meet builders from around the world — not to mention seeing the most incredible robots on the planet.

In the coming months of *SERVO*, I'll focus on how to build a robot in each of the major categories. If you can follow the directions, you'll be able to compete. So warm up those soldering irons, 'cause it's time you stopped reading and started building.

See you next year! **SV**

Hacking a Zip-Zap *for Remote Control*

by Guy Marsden

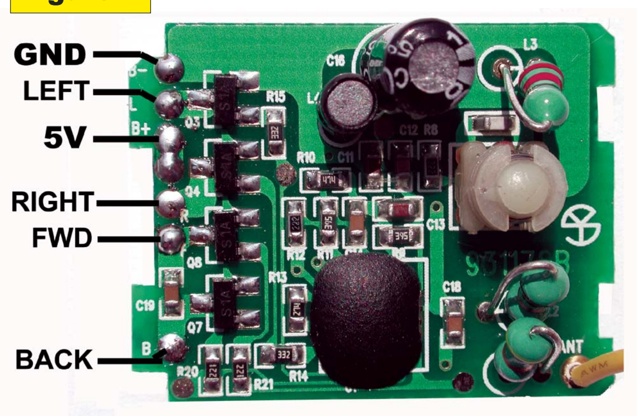
When I was in RadioShack a while back, I noticed that they were selling off the Zip-Zap mini RC toys at deep discounts. The store manager explained that they were closing out the line after a four-year run. (Don't worry — they are still available on eBay and other web resources.) I bought one to indulge my "inner child." I was impressed by the tiny size of the toy car and the fact that it obviously had four channels of control — forward/back and left/right.

After chasing the cat with it for a few minutes and learning that the range was about 10 feet or so, I started to think about applications for a nice, cheap little four-channel control system. A number of ideas came to mind, such as signaling between bots in game playing or flocking behaviors. A transmitter could be used as a sort of lighthouse waypoint for mobile robots that come within its limited range. Or it could be used to keep mobile robots within a certain reception range, such as a specific room.

It could also be used to reduce the number of wires needed to operate manipulators on the end of an arm. Sometimes running multiple wires through a robot arm can be tricky and restrictive; a short-range radio link could reduce a bulky wire harness down to two power leads. So I decided to take my toy apart to see how practical it would be to hack.

Deconstructing the Receiver

Figure 1



On opening up the car, I found a tiny receiver board operating from an equally tiny battery that was marked 1.2V 100 mAH. My first concern was that I would want to get a useful signal level out of the device in order to interface to five-volt logic devices. I removed the battery and clipped in my adjustable bench supply and slowly dialed it up to five volts while operating the transmitter. The car steering (consisting of two tiny electromagnets) still racked back and forth and the motor ran just fine, but much faster. So far so good.

Looking closer at the receiver, I saw a row of SOT-3 parts that were labeled Q3 through Q7, with Q5 and Q8 on the back, so they were likely to be the drive transistors. The solder pads along the edge of the board were nicely labeled: -B, L, +B, R, F, B (see Figure 1). Clearly, these were for power, Left/Right, and Forward/Back. After cutting all the motor wires off close to the board, I soldered some LEDs (with current limiting resistors) to the pads to confirm that the transistors were operating as open collector switches. Success! The transmitter was now controlling four discrete LEDs. (By the way I find it handy to have a bunch of LEDs lying around with 470 ohm resistors attached. They are useful for all kinds of testing and can

be plugged right into a breadboard as an indicator.)

Since the F/B outputs comprise an H-bridge to reverse the motor, it is clear that each output has two transistors apiece that pull the output to the supply rails. This means that a small relay can be connected to one of the outputs and tied to V+ or ground and it will switch when actuated! These outputs can also interface directly to logic since they are driven both ways. The L/R outputs each have a single transistor driver with an open collector so these can switch a relay coil to ground only. To interface to logic, a pull-up resistor would be needed; a 4.7K or 10K will work fine.

Deconstructing the Transmitter

After opening up the transmitter box, I first removed the two AAA batteries (three volts), connected my adjustable supply, and dialed it up slowly starting at three volts while trying the buttons and watching my LEDs blink on the receiver. It worked just fine at 5V, but at 6V it ceased to produce results; the range and response did not seem to be compromised at five volts based on my relative tests. I tried pressing the rubber pads on both the L and R buttons simultaneously hoping to see two LEDs light on the receiver. No such luck — it defaults to just the R output on the receiver. So the system can only operate Left *OR* Right and Forward *OR* Back as I suspected. I was hoping to transmit four bit BCD to make a 16 channel control system, but this way I have the option of transmitting eight discrete values that could be decoded for many uses (See Table 1).

Looking closely at the circuit traces, I noticed that one side of each of the button pads went to ground. I traced the other side of the button pads and found that they all terminated at the 14 pin chip (see Table 2) I attached a color-coded ribbon cable to these points (Figures 2 and 3) and tested them from a CMOS inverter's output to ensure that they would operate the transmitter. This means that interfacing to control logic would simply entail using the output of any logic device or microcontroller pin. It couldn't be a simpler interface.

L	R	F	B	BCD
0	0	0	1	1
0	0	1	0	2
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10

TABLE 1. Transmitter Codes.

IC Pin	CONTROL
1	LEFT
4	FWD
5	BACK
14	RIGHT
3	GND

TABLE 2. Transmitter Connections.

Antennas

It's not a good idea to change the length of the wire antenna attached to the receiver since that would de-tune the RF circuit. However, since the transmitter uses a telescopic antenna, I have to assume that the circuit will be forgiving of adjusting that length. I found by experimentation that anything from a 1" wire to a fully extended telescopic antenna performs about the same. However, as I carried the loose board and battery pack around in my hands, I learned that holding the batteries close to the circuit board (right above or below the components) severely compromised the performance. It would be best to keep batteries and other metal at some distance from a plane parallel to the board; I would think that a metal box would be a bad idea.

Applications

The transmitter board is a tad bulky at 2-1/4" x 3-1/4" since it also incorporates the charging circuit for the car. Since this circuitry is all at one end — near the Forward

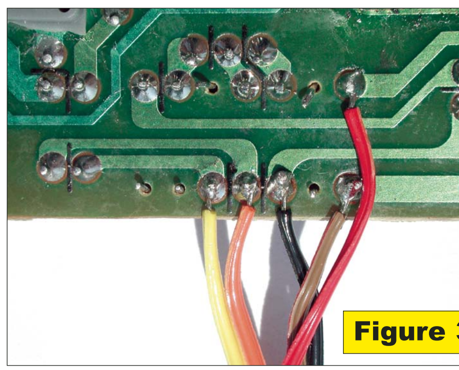
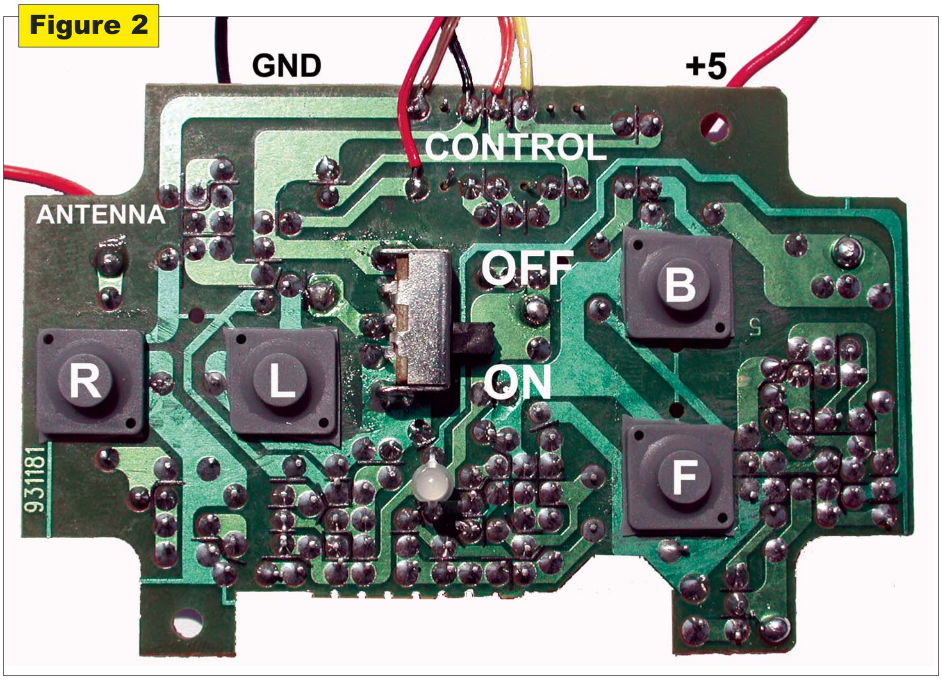
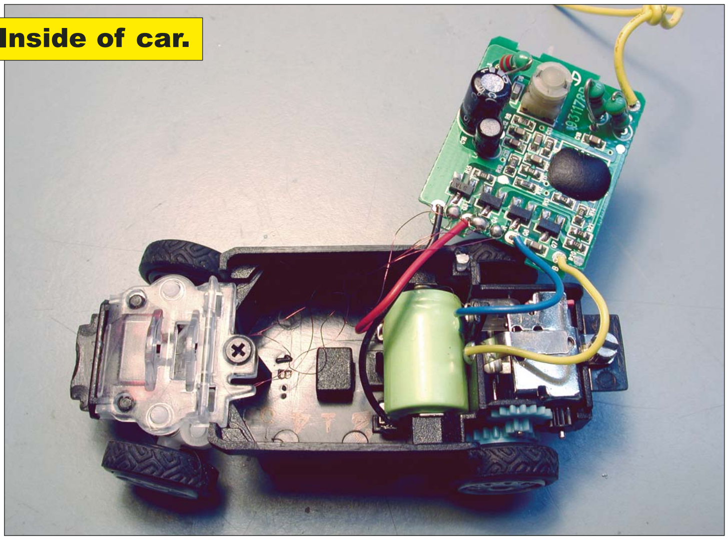


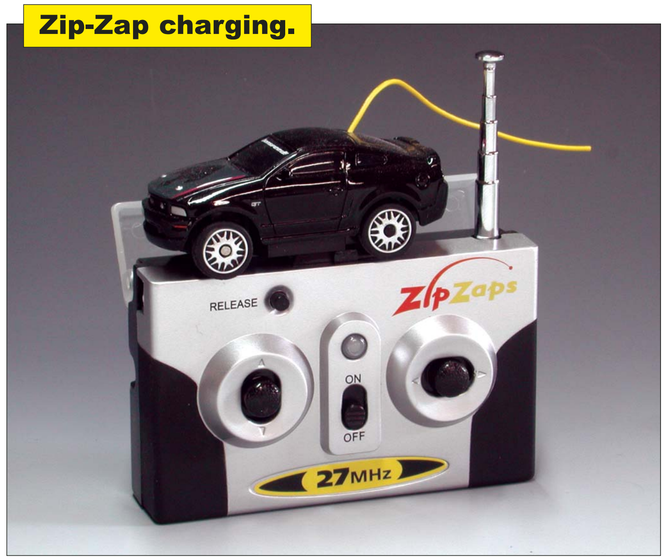
Figure 3

Figure 2





Inside of car.



Zip-Zap charging.

and Back buttons — that part can be carefully cut off if needed. The receiver, however, is pretty darned small at around 1" square and I'm sure that there may be many applications for this tiny four-channel device.

This inexpensive toy can easily be re-purposed in many ways, the applications for the receiver include the following:

- The original use — running one motor forward/stop/back and switching two small loads.

- Switching four small loads or relays.

- Transmitting up to eight discrete codes as a four bit binary coded value.

- A combination of the above like reversing a motor and using the remaining outputs to drive a run/stop relay and a Forward/Back relay for a second motor. **SV**

GEER HEAD

by David Geer

Contact the author at geercom@alltel.net

Max (a.k.a., Ernie) the Ford Showcase Robot

Not A Model, But Not Bad

My Name is Max

Max may not do justice to the latest European bikini, yet exhibition-goers can't take their eyes off of him. Max the exhibition robot (a.k.a., Ernie when in use at Ford auto shows), which has appeared at Ford auto shows as the Ford Robot, is the mechanization of Mannetron, an exhibit robotics company in Battle Creek, MI.

Max is an "anthropomorphic, teleoperated" robot, meaning it has human-like form and function and can be operated remotely via RC technology — in this case, a very sophisticated, operator-worn sensing suit based on virtual reality technology.

As with any performing artist, Max's real work is done "behind the scenes" or apart from the show itself. Max bears the burden of repeated setup, take down, storage, and shipping for show after show around the globe.

To stand up to this routine, Max's materials were carefully selected: brass, aluminum, nylon, and stainless steel. His hands are a combination of lexan, aluminum, and stainless steel. His head and body shell are polyester reinforced by fiberglass (long live the polyester suit!).

This exhibition robot is endowed with a "36-axis, real-time, motion control system." Max's many axis of movement and degrees of freedom enable lifelike interaction with his audiences. Max's many movements break down to "19 aquadraulic motions," aquadraulics being water-based hydraulics, and "17 direct drive electromechanical motions."

Max's voice, which comes from a human operator, is translated by a Roland Voice processor to make it sound like Max every time, whether the operator's voice is high or low, male or female. Two high direction shotgun microphones are used to pick up questions from the audience so Max's operator can hear them and respond.

One of Max's "eyes" is a "head mounted mini hi-res color camera" inside his head. Additional CCTV cameras and monitors — provided by the client putting on the show — give the robot's operator two angle views to aid observation of and interaction with the crowd.

Max the Robot Requires Maximum Construction

Knowing Max must travel the world (US, Europe, Middle East, and Asia) from show to show and endure drop-offs, setup, performance, and

Max the Ford robot, an exhibition robot adapted for the Ford auto shows in this case, winning best of show.

Max side angle, working the crowd, pleasing children young and old.

Max (a.k.a., Ernie) gestures to the ceiling as he strikes a pose.

ROBOT KIOSK

Other Mannetron creations include interactive media stations or kiosks. The kiosks consist of a bright 23-inch LCD screen offering an HD wide screen viewing format, a formidable sound system, a quick response touch sensor, as well as a powerful computer and graphics processing.

take down repeatedly, its creators at Mannetron made sure both Max and its shipping enclosures were equipped to the max!

The base on which Max stands houses all the electrical and hydraulic support systems. Nineteen of his movements are enabled by 22 precision controlled water-based hydraulic pumps. The base also houses "precision DC servo systems," as well as power supplies and two "industrial PCs with 16 DSP motion control sub processors." The base also houses a main power switch and a pressure gauge for the hydraulics.

For shipping, Max is placed on top of his base and covered by a protective cover. All other systems ship in crates and flight cases.

Max's base frame — on which he stands during shows — ships at a weight of 2,200 lbs. Its support equipment, sensor suit, tools for adjustment, maintenance and repair, and spare parts are shipped at a weight of 660 lbs. And, finally, all control system computers and other spare parts and tools are shipped at a weight of 220 lbs. Max, you're a mighty heavy robot.

During performances, Max's base frame is usually hidden by a surrounding structure dedicated to the theme of the particular show. Everything that isn't the robot itself or its frame is housed in a separate control room, constructed on site for the occasion.

Major Max to Room Control

There in the control room, Max's sensor-suited human operator is hidden from view, adding to the impression that Max is alive and acting completely independent of technical support or human intervention.

The body-worn sensor suit measures all robot movements "in real-time at 30 frames per second."

According to Mannetron's project coordinator, Peter Jungen, Max creates a fascination with the crowd by creating "the illusion of a living machine." Max does this by exhibiting fundamental traits of verbal and non-verbal communication as we do.

Max grabs people's attention and holds it by recognizing them, turning towards them, looking at them, saying hello, and responding to questions.

The sensor suit makes all this possible. The suit tracks all Max's body and head movements. Max's head mounted eye (camera) tracks in tandem with the remote operator, who can see what the camera sees through virtual reality goggles. As the operator moves and responds to the audience, so does Max.

The sensor suit has aluminum mechanical parts that parallel the operator's arms. The sensors collect information about the operator's movements and translate them into Max's movements. The suit does all this without restricting the operator or making her/him uncomfortable.

According to Jungen, most of the suit's sensors are "conductive plastic rotary or linear sensors made by Midori (Japan)." Even Max's finger movements parallel those of the operator. This is enabled by flex sensors in Max's sensor gloves.

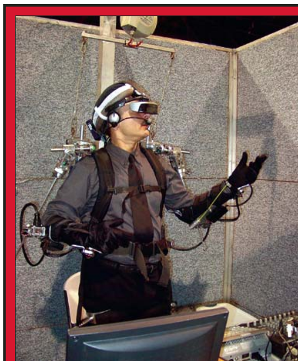
Max's upper body movements are measured by "solid-state gyros." Head motion is tracked by a "three axis inclinometer/magnetometer module made by crossbow." "The jaw motion is generated from the operator's voice pickup using a custom designed tracking circuit," says Jungen. No detail of human motion has been left unconsidered.

A PC loaded with Syncon (Mannetron's robot performance control system) processes all input signals

How do they sell any cars or car ideas with Max holding the center of attention? I see one guy looking at a car while everyone else is studying Max.

Ernie when at the Ford shows is the Mannetron robot pictured in his guise as the Ford robot here.

Mannetron's sensing telemetry suit with human operator shows virtual reality goggles, mic and headset, aluminum arm, leg, and body sensing equipment and sensor gloves.



from the suit in real time using "digital filtering, scaling, and limiting." Any of Max's motions can be captured, recorded, and played back, as well, though live performance is more real and preferred.

The sensor suit and control system in the make shift control room are connected to Max via a 300-foot stretch of cabling. The suit includes a microphone, permitting the operator/wearer to be Max's voice.

In the control room with the operator and suit, you will find Max's audio mixer, which can be connected to a power amp or another mixing console, using XLR outputs.

Along with the 300-foot cable stretched from the control room to Max, are two audio cables so you can transmit the "line level signal" to the robot.

Max Got the Power

To keep the robot's various systems from failing, power output to the robot needs to be reliable. It can't have line spikes, either. So, Max needs to run on his own circuit. Power needs to be provided both to the control room and to the robot, as well (30 amps to both locations running on 110 VAC at 50-60 Hz).

Again, the main cable from the control room to the robot also contains coaxial cables that can be used to connect the cameras and the control room.

Max can also be programmed to run independently through a series of preset movements. In this way, it can capture the audience with gestures and pre-recorded speech or a song and dance number. While Max needs little or no break time, the operator does, and this programmed segment permits them just that.

Max is farmed out for exhibits and marketing purposes with an able technician, ready to help you use the technology with relative ease.

Helping Max Keep His Cool

Though Max doesn't need breaks, he does get hot and needs to keep his cool. Airflow is provided between the stage and frame by fans mounted underneath the robot. Fans should provide at least 500 cfm of airflow. But, if the normal temperature is above room temperature, an air conditioner will be needed, as well. SV

ROBOT CONTROL SYSTEM UPDATE
Syncon VI Robot Performance Control

Max and other Mannetron robots are controlled by Mannetron's Syncon control system. The system has recently been updated to Syncon VI. Whether lifelike "swimming whales, roaring dinosaurs, or Max himself, these animations are produced by the latest in top notch control system technology.

Take a 30-foot robotic T-Rex dinosaur, for example. Assuming full animation and the illusion of real life, the robot has to be capable of many movements requiring massive drive motors. These motors would have to enable many tons of force. Such a robot would also require very small, intricate, and precise movements, "such as a nostril flare or eye blink."

The new Syncon VI ties all these movements together, controlling them to create an orchestrated animation simulating a living creature.

RESOURCES

www.mannetron.com
Parents to Max and numerous anthropomorphic and other robots.

http://mannetron.com/downloads.htm
Several cool videos of Mannetron robots in action.

http://mannetron.com/anthro.htm
Anthropomorphic robots.

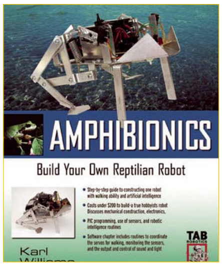
http://mannetron.com/rollaround.htm
Mannetron's roll-around robot greeter.

http://mannetron.com/humanoid.htm
Mannetron's advanced humanoid robotics.

Mind Candy For Today's Roboticist

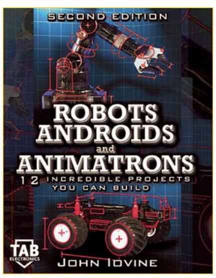
The *SERVO* Store

Amphibionics
by Karl Williams



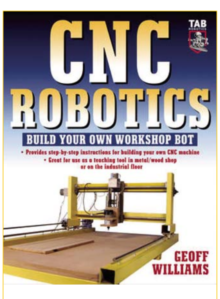
This work provides the hobbyist with detailed mechanical, electronic, and PIC microcontroller knowledge needed to build and program a snake, frog, turtle, and alligator robots. It focuses on the construction of each robot in detail, and then explores the world of slithering, jumping, swimming, and walking robots, and the artificial intelligence needed to make these movements happen with these platforms. Packed with insight and a wealth of informative illustrations. **$19.95**

Robots, Androids and Animatrons — Second Edition
by John Iovine



There's never been a better time to explore the world of the nearly human. In this book, you get everything you need to create 12 exciting robotic projects using off-the-shelf products and workshop-built devices, including a complete parts list. Also ideal for anyone interested in electronic and motion control, this cult classic gives you the building blocks you need to go practically anywhere in robotics. **$19.95**

CNC Robotics
by Geoff Williams

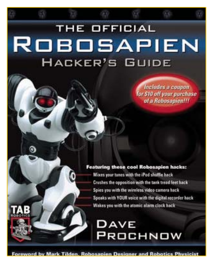


CNC Robotics gives you step-by-step, illustrated directions for designing, constructing, and testing a fully functional CNC robot that saves you 80 percent of the price of an off-the-shelf bot — and that can be customized to suit your purposes exactly, because you designed it. Written by an accomplished workshop bot designer/ builder, this book gives you all the information you'll need on CNC robotics! **$34.95**

SERVO CD-Rom

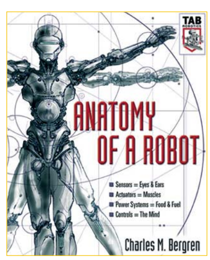
Are you ready for some good news? Starting with the first *SERVO Magazine* issue — November 2003 — all of the issues through the 2004 calendar year are now available on a CD that can be searched, printed, and easily stored. This CD includes all of Volume 1, issues 11-12 and Volume 2, issues 1-12, for a total of 14 issues. The CD-Rom is PC and Mac compatible. It requires Adobe Acrobat Reader version 6 or above. Adobe Acrobat Reader version 7 is included on the disc. **$29.95**

The Official Robosapien Hacker's Guide
by Dave Prochnow



The Robosapien robot was one of the most popular hobbyist gifts of the 2004 holiday season, selling approximately 1.5 million units at major retail outlets. The brief manual accompanying the robot covered only basic movements and maneuvers — the robot's real power and potential remain undiscovered by most owners — until now! This timely book covers all the possible design additions, programming possibilities, and "hacks" not found anyplace else. **$24.95**

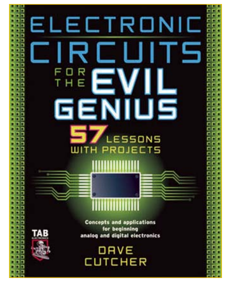
Anatomy of a Robot
by Charles Bergren



This work looks under the hood of all robotic projects, stimulating teachers, students, and hobbyists to learn more about the gamut of areas associated with control systems and robotics. It offers a unique presentation in providing both theory and philosophy in a technical, yet entertaining way. Reading *Anatomy of a Robot* is like having a robot on the operating room table. Crack open the pages and you'll be able to dissect a robot from head to toe. **$29.95**

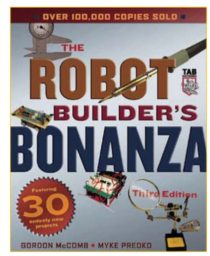
We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

Electronic Circuits for the Evil Genius
by Dave Cutcher



Cutcher's 57 lessons build on each other and add up to projects that are fun and practical. The reader gains valuable experience in circuit construction and design and in learning to test, modify, and observe results. Bonus website **www.books.mcgraw-hill.com/authors/cutcher** provides animations, answers to worksheet problems, links to other resources, WAV files to be used as frequency generators, and freeware to apply your PC as an oscilloscope. **$24.95**

Robot Builder's Bonanza Third Edition
by Gordon McComb / Myke Predko



Everybody's favorite amateur robotics book is bolder and better than ever — and now features the field's "grand master" Myke Predko as the new author! Author duo McComb and Predko bring their expertise to this fully-illustrated robotics "bible" to enhance the already incomparable content on how to build — and have a universe of fun — with robots. Projects vary in complexity so everyone from novices to advanced hobbyists will find something of interest. Among the many new editions, this book features 30 completely new projects! **$27.95**

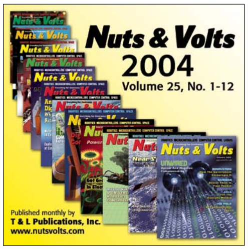
101 Spy Gadgets for the Evil Genius
by Brad Graham/Kathy McGowan

Utilizing inexpensive, easily-obtainable components, you can build the same information gathering, covert sleuthing devices used by your favorite film secret agent. Projects range from simple to sophisticated and come complete with a list of required parts and tools, numerous illustrations, and step-by-step assembly instructions. **$24.95**

To order call 1-800-783-4624 or go to our website at www.servomagazine.com

Mind Candy For Today's Roboticist

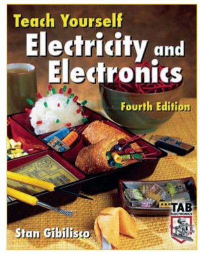
Nuts & Volts CD-Rom



Here's some good news for *Nuts & Volts* readers! Starting with the January 2004 issue of *Nuts & Volts*, all of the issues through the 2004 calendar year are now available on a CD that can be searched, printed, and easily stored. This CD includes all of Volume 25, issues 1-12, for a total of 12 issues. The CD-Rom is PC and Mac compatible. It requires Adobe Acrobat Reader version 6 or above. Adobe Acrobat Reader version 7 is included on the disc. **$29.95**

Teach Yourself Electricity and Electronics — Fourth Edition

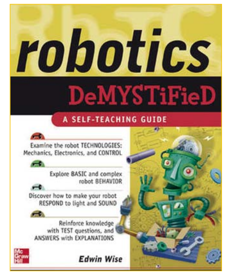
by Stan Gibilisco



Learn the hows and whys behind basic electricity, electronics, and communications without formal training. The best combination self-teaching guide, home reference, and classroom text on electricity and electronics has been updated to deliver the latest advances. Great for preparing for amateur and commercial licensing exams, this guide has been prized by thousands of students and professionals for its uniquely thorough coverage ranging from DC and AC concepts to semiconductors and integrated circuits. **$34.95**

Robotics Demystified

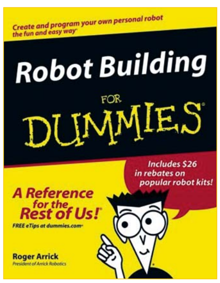
by Edwin Wise



There's no easier, faster, or more practical way to learn the really tough subjects. McGraw-Hill's Demystified titles are the most efficient, interestingly written, brush-ups you can find. Organized as self-teaching guides, they come complete with key points, background information, questions at the end of each chapter, and even final exams. You'll be able to learn more in less time, evaluate your strengths and weaknesses, and reinforce your knowledge and confidence. **$19.95**

Robot Building for Dummies

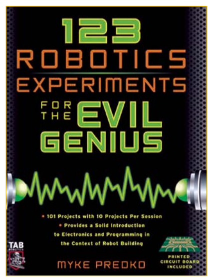
by Roger Arrick / Nancy Stevenson



Ready to enter the robot world? This book is your passport! It walks you through building your very own little metal assistant from a kit, dressing it up, giving it a brain, programming it to do things, even making it talk. Along the way, you'll gather some tidbits about robot history, enthusiasts' groups, and more. Within this book you get explanations in plain English, "get in, get out" information, icons and other navigational aids, tear-out cheat sheet, top ten lists, and a dash of humor and fun. **$21.00**

123 Robotics Experiments for the Evil Genius

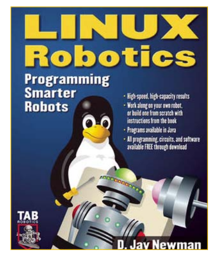
by Myke Predko



If you enjoy tinkering in your workshop and have a fascination for robotics, you'll have hours of fun working through the 123 experiments found in this innovative project book. More than just an enjoyable way to spend time, these exciting experiments also provide a solid grounding in robotics, electronics, and programming. Each experiment builds on the skills acquired in those before it so you develop a hands-on, nuts-and-bolts understanding of robotics — from the ground up. **$25.00**

Linux Robotics

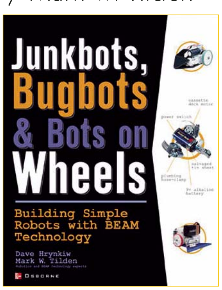
by D. Jay Newman



If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need, plus complete training in doing original programming. **$34.95**

JunkBots, Bugbots, and Bots on Wheels

by Dave Hrynkiw / Mark W. Tilden



From the publishers of *BattleBots: The Official Guide* comes this do-it-yourself guide to BEAM (Biology, Electronics, Aesthetics, Mechanics) robots. They're cheap, simple, and can be built by beginners in just a few hours, with help from this expert guide complete with full-color photos. Get ready for some dumpster-diving! **$24.99**

Check out our online bookstore at www.servomagazine.com for a complete listing of all the books that are available.

$20 each 15% off if you buy two or more.

From HomoSapien to RoboSapien

Before R2D2 there was R1D1

SERVO Magazine T-Shirts

Get your very own limited-edition *SERVO Magazine* T-shirt. Shirts come in sizes S, M, L, and XL, and are available in either black or white. All shirts are 100% pre shrunk cotton.

ROBOTICS RESOURCES

Tune in each month for a heads-up on where to get all of your "robotics resources" for the best prices!

BY GORDON MCCOMB

Soldering for Robotics and Electronics

Even simple robots need some custom wiring, and many require their own electronic circuitry. There are many methods to wiring a robot and building its circuitry, but one of the most common — and therefore most critical — is soldering.

Soldering is where you assemble wires and components and build a permanent electrical system. Instead of tape or rubber bands or other temporary fasteners or glue to hold things together, you use small globs of molten metal. The metal not only provides a physical joint between the wires and components of your circuit, it supplies the necessary conductivity to allow the circuit to work.

Despite working with temperatures exceeding 700 degrees Fahrenheit, soldering is fun, is generally safe (if you observe the normal precautions), and it allows you to build permanent, professional-looking circuits. You need only a minimum of tools and supplies, most of which can be purchased locally, including hardware and home improvement stores.

When to Solder

Before getting into the how's of soldering, let's talk about the why's. Not all electrical circuits need to be soldered. There are easier methods than soldering that you can use to construct a working circuit, especially if all you're doing is experimenting. You're better off using a solderless breadboard if:

- *You are just playing around with ideas*. You'll want to pull components out and try new ones, or experiment with different ways to connect things. While you can solder your circuit designs, any mistakes or changes require unsoldering and therefore a lot of additional work.

- *You're testing your circuit to be sure it works properly*. Even the best electronics experts try out their ideas before committing them to permanent soldered status. With a solderless breadboard, you can more readily make changes to improve the circuit.

- *You don't need or want a permanent circuit*. A simple test circuit for your robot can be built on a solderless breadboard. It's a temporary circuit you don't intend to keep.

- *You want to customize the circuit as you work with it*. Rather than build several options into one circuit, you might want to reconfigure it to change its behavior. This can often be done by switching out basic components, like resistors and capacitors. Such changes can be made in seconds with a solderless breadboard.

On the other hand, soldering is a near necessity for any circuit that requires permanence, or that might be damaged by ordinary handling. You will want to consider soldering if the circuit is exposed to lots of handling, motion, or vibrations that might work the connections loose. This applies to most robotic endeavors, which is why soldering is the more common construction technique.

A properly soldered circuit will last much longer than one mounted on a solderless breadboard. If you plan on using the circuit for more than a few weeks, it should be permanently soldered.

Also important for circuits that rely on microcontrollers or other high-speed electronics is that soldered circuits are less prone to the effects of stray capacitance. Stray capacitance can affect the operation of circuits in unpredictable ways, and is most noticeable with circuits that already rely on capacitors for signal timing. The stray capacitance is caused by long lead lengths on the components, and the construction of the solderless breadboard itself.

What You Need for Soldering

Only basic tools are required for soldering. You can purchase a basic, no-frills soldering setup for under $10, but the better soldering tools will cost you more. Frankly, I recommend the best soldering tools you can get, though for hobby use there's no sense in paying more than $50-$100 for the whole setup.

Soldering Pencil

A soldering pencil is a wand-like tool that consists of an insulating

handle, a heating element, and polished metal tip. It's called a "pencil" because it resembles a writing instrument. The more generic soldering iron takes many forms, including a large gun-like appliance that was common in the 1940s through 1960s. These big soldering irons should not be used with modern electronics, as they produce way too much heat. For standard electronics work, you want a soldering pencil rated at 25-35 watts. Get a soldering pencil with a replaceable tip.

Soldering Pencil Stand

The better soldering pencils come with a stand, but many low-cost ones do not. You'll want to add one if your soldering pencil is stand-less. A stand holds the hot soldering pencil when you're not using it, and it helps prevent accidents. You really don't want the hot soldering pencil to roll off the desk and onto your lap.

Solder

Solder is the soft metal that is melted by the heat of the soldering pencil. The ideal solder for working with electronics is called 60/40 rosin core. This means that the solder contains 60% tin and 40% lead (the exact ratio can vary a few percentage points), and has a core of rosin flux. The flux helps the molten solder flow around the components and wire, and assures a good joint. Solder comes in various diameters; 0.062" diameter is common, but 0.031" is better suited for detail work. Thicker solder is harder to use on small circuits.

Note that soldering releases toxic fumes. Lead-free solder is available if you are concerned about the effects of lead poisoning. These solders contain other mixtures of metal, such as 95% tin and 5% antimony. Even lead-free solders should be treated with care. Many metals are toxic to one degree or another.

Do not use silver solder or any other solder not specifically intended for electronics. These solders may not provide the same conductivity as standard 60/40 rosin core, and they may cause damage to your electronic components, as well as your soldering pencil.

Additional tools useful for soldering include:

- *Wetted sponge* — for wiping off excess solder and flux from the hot tip of the soldering pencil. Just a basic (but clean!) kitchen sponge will do. In a pinch, you can fold up a paper napkin, get it wet, and squeeze out all the excess water.

- *Solder sucker* — for removing excess solder. The sucker is a spring-loaded vacuum. To use, melt the solder you wish to remove, then quickly position the sucker over the molten glob. Activate the sucker, and the extra solder is removed.

- *6X magnifying glass* — for inspecting your work. After soldering, always check your solder joints to ensure that they are clean and well-formed, and that no solder touches adjacent wires or circuit board pads.

- *Rosin flux remover* — bottle or spray can. After soldering, clean any remaining flux to prevent it from oxidizing your circuit.

- *"Third hand" clamp* — Soldering would be a lot easier if everyone had three hands. Failing that, the next best thing is a small weighted clamp that holds the world while you solder. These are commonly referred to as "helping hands" or "third hand." You can purchase them with or without a built-in magnifying glass.

Picking Just the Right Soldering Pencil and Tip

You've already read about the basic soldering pencil you need for electronics work. This tool is composed of a removable tip, a 25-35 watt heating element, and a stand. The basic soldering pencil will get you soldering your circuits together, but not in style. Though they cost a little more, a soldering pencil with an adjustable temperature control yields a better result.

While some of the higher-end variable temperature soldering pencils come with a digital readout that shows you the actual temperature at the tip, for the most part this feature is not needed for basic electronics work. With some practice and experience, you will be able to gauge the proper temperature to use.

You should also select a soldering pencil that comes with a grounded cord and plug. Static discharge can damage certain electronic components, and this static can be generated in copious amounts during soldering. The grounded electrical plug is also considered safer, in the event the soldering pencil comes into contact with a live electrical circuit.

The soldering tip is attached (usually screws on) to the end of the heating element. The tip is what does the actual soldering. There are literally hundreds of soldering tips to choose from, but don't let that confuse you. For most electronics work, you'll want a small conical or chiseled tip.

These come in various tip sizes: 3/64" through 7/64" are useful for most electronics work.

Soldering tips are seldom interchangeable among brands of soldering pencil, even different models by the same manufacturer. Be sure to purchase the correct tip for your make and model of soldering pencil.

Soldering tips should be replaced as they show signs of wear. Look for corrosion, pitting, or plating that is peeling off. Promptly replace tips that are no longer providing adequate heat.

How to Solder

Before soldering, make sure you have all your tools within easy reach. Dampen a small sponge or a folded-up paper towel. Squeeze out any excess water. You want it to be damp, not soaked. Place the soldering pencil securely in its holder, and plug it in. If yours is the adjustable type, turn the heat to approximately 675-750 degrees.

Wait for the tool to reach proper temperature — usually within 60 seconds for most 25-30 watt soldering pencils. Many soldering pencils with a temperature sensor will show when the proper temperature has been reached by lighting or blinking an indicator.

Handy tip for your new soldering tip: Before soldering, "tin" the new tip by heating up the pencil to full temperature and applying a small amount of solder to the tip. Wipe off any excess with the moistened sponge or towel. Periodically use this same technique to keep the tip clean. You can also purchase soldering tip cleaners if dirt becomes caked on and will not come off during regular tip re-tinning.

Successful soldering requires following some simple rules, and lots of practice. Keep the following in mind as you solder:

- The cleaner the metal surface, the better the solder will stick to it. You can clean etched circuit boards and wire ends with isopropyl alcohol. Let dry before soldering.

- Hold the soldering pencil at a 30-45 degree angle to the surface of the work.

- Always apply the heat of the tip to the work, not to the solder. If soldering a wire into the hole of a circuit board, for example, touch the tip to both the wire and the pad. Wait a few seconds, and apply solder to the heated area. Immediately remove the heat once the solder flows.

- Apply just the right amount of solder: too little and the connection will be weak; too much and the solder may form globs that can cause short circuits.

- Avoid applying more solder to an already-soldered joint. This can cause what's known as a cold solder joint.

Be mindful that most electronic components can become damaged if they are exposed to prolonged or excessive heat — five to six seconds is about it. Apply the soldering pencil only long enough to heat the work for proper soldering — no more, no less. When soldering electronic components that are very heat-sensitive, use a clip-on heatsink. Clip the sink to the wire you are soldering, as near to the component itself as you can. The sink will draw off heat, and will help prevent the component from being destroyed. Of course, you must still exercise caution even when using a heatsink.

The Bane of Soldering: Cold Joints

A cold solder joint is the result of solder that has not properly flowed around the metal parts. Cold joints are physically weaker than properly made joints, and they do not conduct electricity as well. Cold joints can often (but not always) be identified just by their looks. A cold joint typically has a dull rather than shiny appearance, and the solder may form jagged peaks rather than an all-around smooth surface. Cold joints are caused by insufficient heat or metal flow when soldering. A good solder joint will have a uniform, bright metal appearance.

A cold solder joint can be caused by many things. Most common is the work gets moved as the solder cools. Avoid all movement until the solder hardens. Solder or surfaces that are dirty or oily can reduce joint quality. Be sure all metal-to-metal contact is clean. Cold joints can also be created if the work is not heated to the proper temperature. Be sure the work is hot enough so that the solder melts to a somewhat runny liquid. The same effect occurs if the solder is directly applied to the soldering pencil, and not to the heated work.

To fix a cold joint, you will need to unsolder it and discard all the old solder. Avoid reusing the solder — applying fresh solder yields the best results.

Reducing Static Discharge While Soldering

The soldering process can generate electrostatic discharge (ESD), which can cause damage to sensitive electronics components. The static can result from simple manual handling of the components and circuit board, as well as from the soldering pencil itself. You cannot totally eliminate static discharge, but you can minimize it.

Not all electrical components are static sensitive, though for safety sake you should develop "static safe" work habits for all the components you handle. At the bottom of the risk scale are resistors, capacitors, diodes, transformers, coils, and all passive components, such as batteries, switches, and con-

PCB MAKERS

You can design your own printed circuit boards rather than using generic pre-etched and pre-drilled prototyping boards. From there, you can manufacture the board yourself or send it out. I vastly prefer the latter: it's easier, cleaner, and less toxic. It may even be less expensive, depending on the type and quantity of boards you are making. Following are just a handful of companies that will make finished printed circuit boards from your submitted designs. Several of these companies advertise in *SERVO Magazine*; be sure to check them out.

ExpressPCB
www.expresspcb.com

EZPCB
www.ezpcb.com

PCB Cart
www.pcbcart.com

PCB Fab Express
www.pcbfabexpress.com

PCB Pool
www.pcbpool.com

Pulsar, Inc.
www.pulsar-inc.com

R4 Systems, Inc.
www.labcenter-electronics.com

nectors. These are relatively immune to ESD damage unless you happen to be Zeus. Take more care when using TTL integrated circuits and bipolar transistors, and exercise special caution with any CMOS or similar based transistor, integrated circuit, or microcontroller.

What you wear can greatly impact the amount of static that is developed around you. Wool clothing tends to generate static. Instead, wear natural cotton or synthetics. If you're working indoors over carpet, wear regular street shoes instead of going barefoot or in socks. Wear an anti-static wrist band whenever possible. A wire from the wrist band attaches to any grounded object, and helps to draw off static from your body.

Some types of carpets are more prone to static than others. Regular nylon carpet can generate copious amounts of static as you walk over it. Be sure to drain the static from your body — touch a doorknob or the metal of a grounded appliance — before touching anything else. The fiber in many types of commercial carpeting is made to be low-static. Consider this type if you need carpeting for your workroom. You can buy small remnant pieces of low-static or anti-static commercial carpeting from a carpet dealer.

Static buildup can be a serious problem in dry weather. If you live in a dry climate, you'll need to take extra precautions against ESD. Humidifiers, electrically grounded anti-static mats, and other ESD control products are available to reduce static, but the cost for these can run into serious money.

FIGURE 1. ExpressPCB manufactures finished printed circuit boards using your own submitted design.

Unsoldering and Resoldering

Even the experts sometimes insert a component backwards! It's inevitable that a solder joint will need to be undone to fix mistakes, or to clean up a cold solder joint. When this happens, the solder at the joint needs to be removed, and new solder applied. You can use a desolder pump, solder wick, or both to remove solder from the joint. I prefer the solder pump; the solder pump works by sucking up the excess solder with a vacuum. Solder pumps come in two basic styles: spring-loaded plunger and bulb.

With a bulb desolderer, you squeeze the bulb to suck up the solder. These are harder to use unless the bulb is mounted to the soldering pencil. Some soldering pencils especially designed for desoldering are available in this arrangement.

Handy tip: Use solder wick (also called solder braid) to remove hard-to-reach solder. The wick is really a flat braid of copper. It works by being more "absorbent" of solder than the tin plating of most components and printed circuit boards. Exercise care when using solder wick, as touching the hot braid can cause serious burns. With the old solder removed, you may reapply solder to the joint.

Soldering Tips and Techniques

Soldering is not particularly difficult. You'll get the hang of it soon enough, and your work will improve the more time you take to do it right. We'll close this month's column with a list of various tips, techniques, reminders, and suggestions you'll want to consider when soldering:

- Store your spool of solder in a resealable plastic bag. This will help keep it clean. It may pick up dirt and oils if you simply throw it into the toolbox. If the spool does get soiled, clean it with isopropyl alcohol before using it again.

- Cleanliness is king. Be sure all surfaces to be soldered are free of dirt and oils. Otherwise, the soldered joint may be weak, or conductivity could be impaired.

- Allow the soldering pencil to cool completely before putting it away. If you don't use the soldering pencil often, put it in a large plastic bag to keep it clean.

- If the electrical cord of your soldering pencil is grounded, be sure to plug it into a grounded outlet. Don't cut off the ground connector, or bypass the grounding using an adapter. The ground is there for safety, and to help

dissipate static discharge.

• Surplus dental picks make for good soldering tools. You can use the picks to clean the work area prior to soldering, and to scrape away excess solder from a joint. You can get used (but cleaned!) dental tools from a variety of mail-order surplus, including American Science & Surplus (**www.sciplus.com**).

• After soldering, and when you're sure your circuit is operating properly, spray or brush on some flux cleaner. This chemical removes the leftover rosin flux.

• The same general techniques are used to solder surface mount components. With practice, a steady hand, and a good eye (or a good magnifying glass!), you can solder many types of surface mount components. Don't try this if you're new to soldering. Get some experience under your belt first.

Sources

The following online sources provide soldering tools and supplies, including pre-etched and pre-drilled general-purpose boards that you can use to build your own robot circuits.

All Electronics
www.allelectronics.com
General solder tools, supplies. Offers a selection of pre-etched and pre-drilled prototyping boards.

Circuit Specialists
www.web-tronics.com
Circuit Specialists has a very good selection of solder tools and supplies, prototyping boards.

Electronics 123
www.electronics123.com
Electronics 123 provides mostly kits and electronic components, but a good selection of high-quality soldering stations and supplies.

Jameco
www.jameco.com
Large mail order company specializing in electronic parts and supplies, for both hobbyists and pros. I prefer to use their printed catalog, but you can also shop online.

Halted Electronic Supply
www.halted.com

Primarily a surplus mail order company, but they also have a large assortment of new soldering tools, replacement tips, and various soldering supplies.

Mouser Electronics
www.mouser.com

Very large "stocking distributor" of all things electronic. Good source for higher-end soldering stations and tools.

Ramsey Electronics
www.ramseyelectronics.com

Though mostly known for their kits, Ramsey also provides all the soldering tools and supplies you need to build them — and most anything else.

C&S Sales
www.cs-sales.com

Among other types of tools, C&S sells soldering stations, from budget models to higher-end units. **SV**

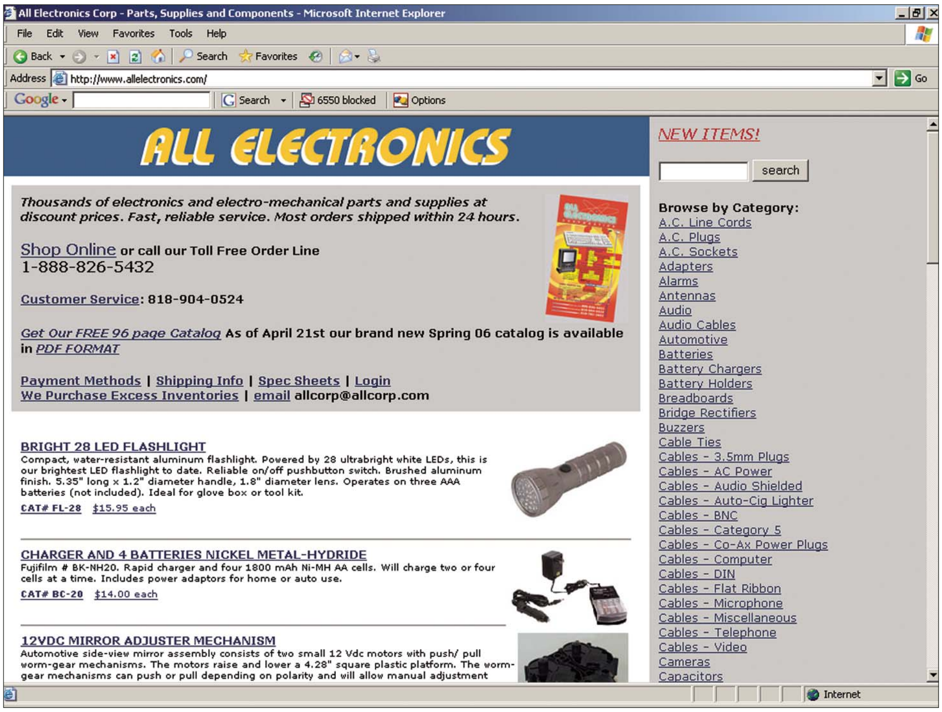
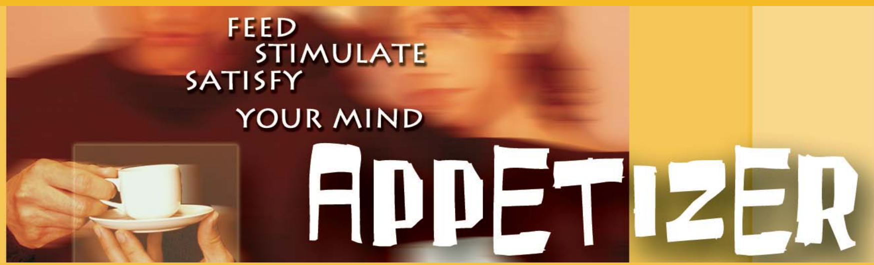


FIGURE 2. The All Electronics website lets you search by keyword (e.g., "solder") or by category.



What Are We Waiting For?

by R. Steven Rainwater

In the late 1940s, neurophysiologist Grey Walter constructed what many consider to be the first modern autonomous mobile robots. He called his homebrew robot design Machina speculatrix and named two of the robots Elmer and Elsie. Most people today remember them by their more common name, tortoises, because of their shape and slow speed. The robots were the result of Walter's intuition that a relatively small number of brain cells could result in complex behavior. His robots exhibited phototaxis, avoided obstacles, sensed their power level, and could find their recharging station when needed. The robots explored their environment by wandering around "speculatively" (thus the name Machina speculatrix). They interacted with their environment using reflex-like behaviors and four modes of operation which he called search, move, dazzle, and touch.

Today, 60 years later, one would expect to see some pretty significant advances among homebrew robots. Modern robots take advantage of batteries with energy and power densities far beyond those Walter used. We have access to a bewildering variety of sensors where Walter relied on only a few simple ones. Materials such as carbon fiber composites, lexan, and titanium are readily available to the hobbyist.

Despite all these technical advances, it's rare to see a hobby robot that can match the behaviors of Walter's tortoises, much less exceed them. And if you want to extend your definition of the word robot far enough to include battlebots, you could even argue that hobby robots have regressed significantly since Walter's time.

Nikola Tesla, after all, demonstrated his remote controlled teleautomaton in 1898. It seems that, at best, we've been building variations on Walter's robots over and over again for decades.

This question of why robots aren't getting smarter came up at a recent meeting of my local robot club, the Dallas Personal Robotics Group. DPRG members have spent a lot of time building robots since the 1980s and we seem to spend almost as much time sitting around in the evenings debating the philosophical underpinnings of our hobby.

Our discussion brought up a number of possible answers to the question of what's holding up the advancement of robots. The discussion left me thinking about what specific goals I could suggest to a robot builder if the desire is to achieve smarter robots.

First Goal — Pick a microcontroller and programming language you're comfortable with and stick with them. Develop a body of reusable code that you can take from one robot project to the next, always improving it, rather than starting from scratch each time. I've observed many robot builders who constantly change from one microcontroller to the next, often because they believe what's needed for a smarter robot is a faster processor. Available computational power far outpaces our ability to use it. Elmer and Elsie had only tiny analog computers Walter constructed from vacuum tubes, yet they would fare pretty well against most modern hobby robots I've seen. Any hobbyist can afford a processor equal in computational power to a small insect brain, which exhibits a level of intelligence far beyond most of today's robots. Yet, most hobby robots are barely at the intelligence level of an amoeba. Creating one that could match wits with an ant would be quite a feat.

DPRG member David P. Anderson offered a music analogy for this principle. Those who argue that we need a faster processor "suggest we are waiting on Cristofori to invent the piano, where it seems to me that we are waiting on Mozart and Beethoven to show us how to play it." Until a Beethoven of robotics comes along, however, the best way we can learn to use the hardware we've got is by sticking with it and making improvements to our software designs. David's music analogy stretches a bit further to suggest that it's better to play one instrument well than several badly.

This first goal may not apply to you if what you're after isn't smarter robots. If your desire is to learn about microcontrollers and programming languages, it probably makes sense to try out lots of different hardware and rewrite the same software over and over in different languages. But don't fall into the trap of thinking that switching to a new board will solve all your problems, because it usually won't.

Second Goal — Add more sensors to your robot. How many sensors does a robot need? More. If you compare your robot to even the simplest forms of life — say a one-celled animal like an amoeba or a multi-celled animal like a nematode — you'll notice right away that your robot is essentially deaf, dumb, and blind compared to biological life.

I believe this point is very important, so let's spend a moment thinking about it. Behavior is an interaction with the environment. In order to interact, a robot has to sense the environment. A typical hobby robot might have a few IR or sonar proximity detectors, maybe some bump sensors or wheel encoders. Surely that's more sensory hardware than a one-celled animal could have, right? Wrong. One-celled animals have an amazing range of senses that allow for an equally impressive range of behaviors. They can sense light, moisture, touch, temperature, a variety of chemical compounds, electrical currents, acceleration, magnetic fields, angular velocity gradients, current direction (in a fluid), rigidity gradients (how hard or soft a surface is), to name a few.

If you want the behavioral complexity of an amoeba, plan on a minimum of 10 types of sensors. I haven't studied insects enough to venture a guess on how many types of sensors they have, but I suspect it's at least an order of magnitude greater than a one-celled animal. Can your robot sense temperature, light, and sound? Does it have any type of inertial sensors? Don't forget to give your robot sensors to monitor its internal states, as well. Does your robot know its battery voltage, the current draw and rotational speed of its motors, the CPU load?

Third Goal — Learn from nature. The first two goals hint at this. Let's make it explicit. If you're trying to make a smarter robot, studying animals is a good way to find ideas. At our current level of technology, the best types of animals to look at are protozoa, nematodes, and insects. We have the hardware, sensors, actuators, and materials to make a robot that's as smart and behaviorally complex as any of these animals.

We aren't there yet and it's a very hard problem. The main thing we lack is a complete understanding of what's required to make a brain. Subsumption architectures and the earlier cybernetic feedback approach used by Walter have proven to be good ways of creating complex behavior from simple reflex-like interactions with the environment. To go further, we need to understand what brains do. An amoeba doesn't have a brain. A nematode has a nervous system composed of a few hundred neurons, but, in general, we don't call it a brain. Insects seem to be the simplest form of life that we refer to as having a brain.

We need to look at how insect brains solve the problem of representation if we want to move our robots a step closer to insect-level intelligence. Having mental representations of the environment is an integral part of higher level intelligence. Unfortunately, it has also become a political issue among researchers.

For decades, AI research concentrated on high-level symbolic representation in machines that had almost no interaction with the real world. Despite plenty of profitable technological spin-offs, they never solved the real problem of creating artificial intelligence. Modern researchers like Brooks concluded that disembodied symbolic thinking machines were not the right approach to achieve intelligence.

Brooks came up with subsumption, in essence, taking us back to cybernetics and Walter's tortoises, which rely on tight interaction with the real world instead of an abstract model of it. Brooks goes a step farther though, eschewing representation altogether, to the extent of talking about the need for "abstraction barriers" that prevent any sort of centralized representation.

Representation appears not to serve any explanatory role in the behavior of animals without brains, such as the amoeba and nematode. In insects, however, it begins to serve an explanatory role. Some insect behaviors are based, not on direct interaction with the environment, but with a mental representation in their brain. Insects retain the simpler reflex-like behaviors, as well.

Phonotaxis in crickets is directly linked to sound sensed in the environment and doesn't rely on any apparent mental representation. Some vision-related behaviors in bees and other insects, however, can only be explained as interactions with mental representations inside the insect's brain rather directly with sensory data. Bees are clearly not doing any sort of classic-AI-style symbolic representation — they aren't necessarily "thinking" about what they see in the sense a human would. They are, however, using a form of representation.

Reflex behaviors alone won't get us to insect level behavior. To get there, we need to learn from nature and find a way of fusing the simple reflex-like ideas of cybernetic feedback and subsumption with new methods of internal representation.

There are plenty of other challenges down the road, but these are some good first steps in the right direction. Let's hope we see the successor to Elmer and Elsie in the near future. **SV**

AUTHOR BIO

R. Steven Rainwater can be reached via email at ***steve@ncc.com***

Then and NOW

ROBOTS WHO SPEAK

by Tom Carroll

One of the first non-mechanical attributes that many of us apply to our robot projects after giving them mobility is a voice. A voice seems to be a great step in bringing the robot from the lowly position of "a robot that ..." to the higher level of "a robot who ..."

Nowadays, we accomplish robot speech by one of two basic ways: computer speech synthesis (which can be accomplished by many techniques) and by recorded voice playback. This is the electronic and computer age. We have at our disposal many interesting ways of creating a human-like voice, but how was it done a century or more ago?

Early Speech Synthesis

The first experimental human speech production by a machine was made in 1773 by G. Kratzenstein, professor of physiology in Copenhagen, Denmark. He was successful in producing vowel sounds by using resonant metal tubes similar to organ pipes. His background in human physiology helped him understand how sounds are produced in our mouth cavity and how the tongue changes these sounds as required for speech.

In 1791, Wolfgang von Kempelen constructed an "Acoustic-Mechanical Speech Machine" in Vienna, Austria. Von Kempelen was an educated man employed by the Empress Maria Theresa in Vienna. He was known for many other experimental ideas such as a mechanical chess player, but the study of human speech was his main area of expertise.

His speech machine was the first mechanical device to produce speech sounds, as well as whole words and short sentences. The three drawings in Figure 1 show the bellows and voice box used to produce sounds. The machine consisted of a bellows that simulated the human lungs that drove air into the "wind box." There was a rubber "mouth" and "nose" that were manipulated by the right arm and fingers to produce vocal sounds.

Varying the length of a vibrating reed changed the tone of the produced voice. Changing the depth of various cavities produced different phoneme, vowel, consonant, and other speech component sounds. Extra bellows, vibrating reeds, and variable cavities combined to produce life-like speech. According to von Kempelen, it was possible for a person to learn to "play the machine" within a period of three weeks, especially if Latin, French, or Italian language was used. He found that his native language of "German was more difficult because of its many closed syllables and consonant clusters."

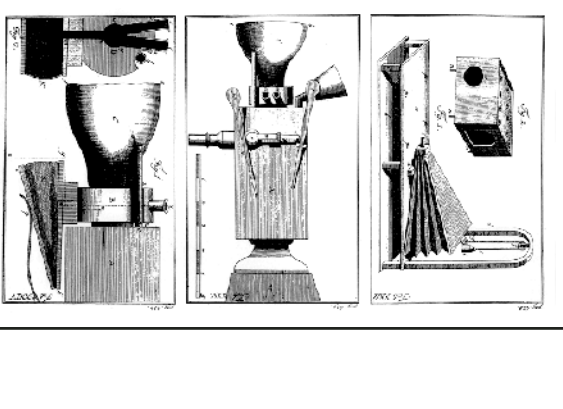
Von Kempelen's speaking machine was first placed in a museum in Vienna and later transferred to the Deutsches Museum in Munich where it is now on display (Figure 2). It can still be operated to this day, though there are parts that are missing or inoperable.

These two men are not the only people to have experimented with speech-producing devices by that time. History has recorded successful attempts at artificial speech using such things as complex "duck calls" to produce a single word back as far as the 10th century and even earlier. "Speaking heads" were made by Gerbert of Aurillac as far back as 1003, by Albertus Magnus, who lived from 1198 to 1280, and by Roger Bacon who lived from 1214 to 1294.

Throughout the 18th, 19th, and well into the 20th centuries, people have experimented with improved mechanical speech producing devices. Speech is something unique only to us

Figure 2. Speech machine in museum.

Figure 1. Von Kempelen's speech machine.



humans and has intrigued mankind since the beginning. We've all seen animals and parrots that seem to speak, but they only repeat phrases that we teach them and do not have a clue as to their meaning. "Polly want a cracker" may get a human response of a cracker but being taught "Polly want's to bite you" and getting a cracker in return results in the same thing. (I hope I didn't offend any of you parrot lovers.)

Thomas Edison's Talking Doll

The other method of speech production is by means of playback of recorded speech. Needless to say, this technique was not available until Edison and others developed the first recordings and phonographs to play them at the end of the 19th century.

In 1890, Thomas Edison decided to expand a bit on his phonograph for the home and produce a doll with a tiny phonograph inside the doll's body. Aimed at the childrens' toy market, this doll was a first of its kind. Figure 3 shows the beautifully crafted doll and the rather crude phonograph mechanism that was inside the body. Note the large wax cylinder that is inscribed with the recorded honograph grooves. The second large cylinder is a drive pulley over which is a rubberband connected to the small flywheel below. The mass of this flywheel kept the speed somewhat constant in the hands of a small child turning the crank. Screw threads were machined onto the crank's shaft that moved the stylus along the fragile wax grooves.

When the crank got to the end of the recording, the screw disengaged and a spring lifted the stylus and returned it back to the beginning. When the miniature phonograph was installed in the doll, the small horn pointed up to holes in the doll's chest. The phonograph mechanism was 7" tall and the wax cylinder was 3" in diameter and 5/8" wide.

Edison built a 40 by 210 foot manufacturing building for his Edison Phonograph Toy Manufacturing Company that had a capacity to make 500 dolls a day, or over 100,000 dolls a year. There were 250 people involved in the production, including a room of ladies whose only job was to record the nursery rhymes on each individual playback mechanism.

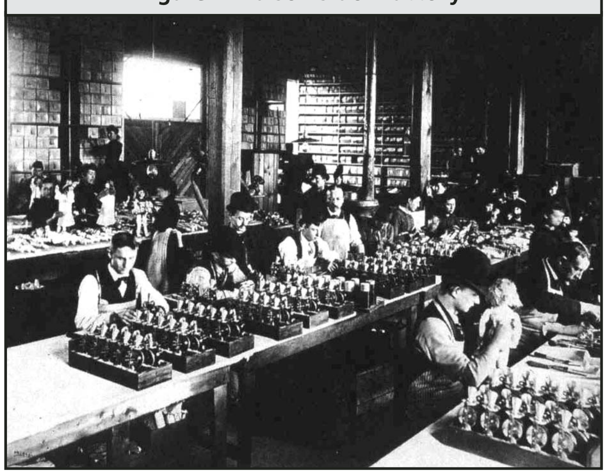
Figure 3. Edison's doll and internal phonograph.

Figure 4 shows the dark factory floor with the workers hand-forming the crude phonograph's mechanisms. Each of the four pound dolls were 22 inches long and cost $10 — a tremendous sum in the late 18th century. This high cost — along with a very poor playback mechanism and poor marketing — caused the whole business to fold.

Uniquely enough, it was actually another person, William W. Jacques, who developed a working prototype from Edison's early tinfoil phonograph and founded the doll company in 1887. Edison later bought him out, demoted him, and a series of lawsuits followed.

It has been said that only one out of a hundred of the dolls actually worked correctly after delivery to a buyer's home, and after a few playbacks of the recording, even these dolls failed. The wax recording was far too fragile and wore out very quickly. There are few records of actual sales of the doll, but recovered files indicate as few as 300 were actually sold. Inventories of the remaining dolls show 7,557 were on hand in 1890, but only a handful of the complete dolls are still in existence today. Someone once told me that one of Edison's dolls went for over $15,000 at an auction — not bad for a toy whose squeaky voice you could barely understand.

Figure 4. Edison's doll factory.



If you think about it, the 300 dolls sold represent only 60% of a single day's production before failure of the business. In my mind, that ranks among the very worst business ventures in history — a large factory built and 250 people hired for just part of a day's production! You can hear a playback of the doll at **http://almost-something.com**

Homer Dudley's Voder

As the telephone replaced the telegraph and communications became very important for businesses and daily life in the first part of the 20th century, research into bandwidth concerns became important. Homer Dudley's Vocoder (VOiCe Operated recorDER) in the late '30s was the first serious attempt at using electronics to analyze speech for telecommunications purposes.

The Vocoder analyzed incoming speech using band-pass filters and

Figure 5. Woman playing the Voder.

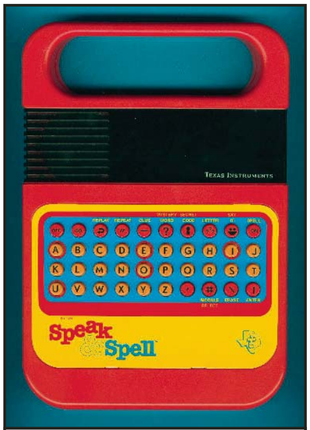


Figure 6. Speak and Spell.

the resulting time-variant band-level information to filter synthetic sounds from a series of oscillators with a matching bank of filters. The earliest music synthesizers used his technology for the new type of music. Dudley used similar technology to build a keyboard-controlled speech synthesizer — the Voder — which is shown in Figure 5.

This extremely difficult instrument to "play" was a huge hit at the 1939 World's Fair, the same fair that introduced Westinghouse's Elektro the Robot to amazed audiences. It took a person quite a bit of time to learn how to hit multiple keys at the same time while varying other key strikes to produce just a single phoneme. Of course, other sets of keys must then quickly be held down to finish the word. A short sentence could take a hundred key strokes or more to complete.

With the advent of electronic means — band-pass filters, linear-predictive-coding, square-sine-sawtooth oscillators and, of course, computers — electronic music and speech took off. The first computer speech synthesis systems were built in the late 1950s and the first text-to-speech system was built in 1968. Physicist, John Larry Kelly of Bell Labs used an IBM 704 computer to synthesize speech and singing.

Author Arthur C. Clarke happened to be visiting the Bell Labs and heard Kelly's computer singing "Daisy, Daisy." He was so impressed by the demonstration that he used the song to be sung by the malfunctioning HAL 9000 computer that astronaut Dave Bowman lobotomized in the movie *2001 — A Space Odyssey*.

The Texas Instruments Speak and Spell

Texas Instruments created the Speak and Spell during the late 1970s. Speak and Spell was originally advertised as a tool for helping young children to become literate, learn to spell, and learn the alphabet (Figure 6). First sold in 1978, other variations included the Speak and Read and the Speak and Math. If you think about it, how else could a machine teach spelling without having to actually speak the word? It was an ideal application.

TI started the Speak and Spell project in 1976 and used their TMC 0280, the first one-chip LPC (linear predictive coding) speech synthesizer. The Speak and Spell made its film debut in *ET* when ET used part of one of the units to "call home."

Later products used the TMS 5100, 5200, and 5220 Voice Synthesis Processors in commercial products needing synthetic speech voice output from digitally-stored words and phrases. Speech data was stored in up to 16 128K ROM chips (TMC 0350). Car manufacturers quickly jumped on the bandwagon to use these chip sets to warn drivers of situations. We all remember the annoying "The door is ajar" message in cars of 20 years ago.

Nowadays, we have so many speech synthesizers for use in our robot projects. Many use a computer or microcontroller to create the words, yet there are also some stand-alone units that only need a simple machine code "word" to output a word or phrase. There are also electronic voice recorders that can have individual words or phrases be triggered by an external signal.

Jameco, Mouser, and other suppliers listed in *SERVO* and *Nuts & Volts* have all sorts of units from the $20 range to several hundred dollars. I didn't list the many manufacturers and models available, but you can go to Gordon McComb's *Robot Builder's Bonanza* or the Internet and find hundreds of sources. The Internet has so many sources on the history of ynthetic speech. The next time someone says to your robot "speak up," it will ... all by itself! **SV**

Advertiser Index